



PCT/GB2004/004157

PCT GB04/4159

INVESTOR IN PEOPLE

The Patent Office
Concept House
Cardiff Road

Newport REC'D 19 OCT 2004
South Wales
NP10 800 WIPO PCT

PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH
RULE 17.1(a) OR (b)

I, the undersigned, being an officer duly authorised in accordance with Section 74(1) and (4) of the Deregulation & Contracting Out Act 1994, to sign and issue certificates on behalf of the Comptroller-General, hereby certify that annexed hereto is a true copy of the documents as originally filed in connection with the patent application identified therein.

In accordance with the Patents (Companies Re-registration) Rules 1982, if a company named in this certificate and any accompanying documents has re-registered under the Companies Act 1980 with the same name as that with which it was registered immediately before re-registration save for the substitution as, or inclusion as, the last part of the name of the words "public limited company" or their equivalents in Welsh, references to the name of the company in this certificate and any accompanying documents shall be treated as references to the name with which it is so re-registered.

In accordance with the rules, the words "public limited company" may be replaced by p.l.c., plc, P.L.C. or PLC.

Re-registration under the Companies Act does not constitute a new legal entity but merely subjects the company to certain additional company law rules.

Signed

Dated 12 October 2004

Request for grant of a patent

(See the notes on the back of this form. You can also get an explanatory leaflet from the Patent Office to help you fill in this form)

1/77

The
Patent
Office30SEP03 E840776-1 D03052
P01/7700 0.00-0322765.9PATENT OFFICE
H
29 SEP 2003
LONDONThe Patent Office
Cardiff Road
Newport
Gwent NP10 8QQ

1. Your reference

A30294

2. Patent application number
(The Patent Office will fill in this part)

0322765.9

3. Full name, address and postcode of the or of each applicant (underline all surnames)

BRITISH TELECOMMUNICATIONS public limited company
81 NEWGATE STREET
LONDON, EC1A 7AJ, England
Registered in England: 1800000

Patents ADP number (if you know it)

1867002 ✓

If the applicant is a corporate body, give the country/state of its incorporation

UNITED KINGDOM

4. Title of the invention

MULTI-LEVEL MATCHING PROCESS

5. Name of your agent (if you have one)

"Address for Service" in the United Kingdom to which all correspondence should be sent (including the postcode)

BT GROUP LEGAL
INTELLECTUAL PROPERTY DEPARTMENT
HOLBORN CENTRE
120 HOLBORN
LONDON, EC1N 2TE

Patents ADP number (if you know it)

1867007

6. If you are declaring priority from one or more earlier patent applications, give the country and the date of filing of the or of each of these earlier applications and (if you know it) the or each application number

Country Priority application number (if you know it) Date of filing (day / month / year)

7. If this application is divided or otherwise derived from an earlier UK application, give the number and the filing date of the earlier application

Number of earlier application Date of filing (day/month/year)

8. Is a statement of inventorship and of right to grant of a patent required in support of this request? (Answer 'Yes' if:

YES

- a) any applicant named in part 3 is not an inventor, or
- b) there is an inventor who is not named as an applicant, or
- c) any named applicant is a corporate body.

(See note (d))

9. Enter the number of sheets for any of the following items you are filing with this form.
Do not count copies of the same document

Continuation sheets of this form -

Description 43

Claim(s) 7

Abstract 1

Drawing(s) 10 + 10

JML

10. If you are also filing any of the following, state how many against each item

Priority Documents

Translations of priority documents

Statement of inventorship and right to grant of a patent (Patents Form 7/77)

Request for preliminary examination and search (Patents Form 9/77) YES

Request for substantive examination (Patents Form 10/77)

Any other documents (please specify)

2 COMPLETE SETS OF TIME-SLOT ASSIGNMENT PROCESS.

I/We request the grant of a patent on the basis of this application.
Signature(s) Date:

29 September 2003

LIDBETTER, Tomothy Guy Edwin, Authorised Signatory

12. Name and daytime telephone number of person to contact in the United Kingdom

Rod HILLEN

020 7492 8140

Warning

After an application for a patent has been filed, the Comptroller of the Patent Office will consider whether publication or communication of the invention should be prohibited or restricted under Section 22 of the Patents Act 1977. You will be informed if it is necessary to prohibit or restrict your invention in this way. Furthermore, if you live in the United Kingdom, Section 23 of the Patents Act 1977 stops you from applying for a patent abroad without first getting written permission from the Patent Office unless an application has been filed at least 6 weeks beforehand in the United Kingdom for a patent for the same invention and either no direction prohibiting publication or communication has been given, or any such direction has been revoked.

Notes

- If you need help to fill in this form or you have any questions, please contact the Patent Office on 0645 500505.
- Write your answers in capital letters using black ink or you may type them.
- If there is not enough space for all the relevant details on any part of this form, please continue on a separate sheet of paper and write "see continuation sheet" in the relevant part(s). Any continuation sheet should be attached to this form.
- If you have answered 'Yes' Patents Form 7/77 will need to be filed.
- Once you have filled in the form you must remember to sign and date it.
- For details of the fee and ways to pay please contact the Patent Office.

DUPPLICATE

1

MULTI-LEVEL MATCHING PROCESS

This invention relates to a multi-level matching process for use in a contention resolution scheme for cell, packet and circuit switches and networks. In particular, but not 5 exclusively, the invention relates to a scalable hierarchical matching algorithm which matches asymmetric request matrices.

The present invention relates to the matching aspect of the scheduling of cells or packets, buffered at the input ports of a switch, for switching across the switch fabric. The overall 10 scheduling operation comprises the matching process described herein and a time-slot assignment process. One example of a time-slot assignment process is described by the inventors in their patent specification entitled " TIME-SLOT ASSIGNMENT PROCESS", contemporaneously filed with this specification, a copy of which is filed herewith and incorporated into the description by reference.

15

The term switch is used herein to refer to switches and/or routers and/or networks which forward data towards their destination, such as are used in communication networks, for example, the Internet. The present invention also relates to the matching of circuit-switched service requests, such as connections and information rates, for switching 20 across a switch fabric. This description is written in terms of cell and packet switches, but the principles also apply to circuit switches (for example, in the context that the matching process seeks to grant service requests without contention, and the services requests can equivalently be requests for bandwidth etc. in a circuit switch).

25 As communication networks, particularly the internet, evolve, faster and more efficient switches are needed, for example, switches capable of exceeding Terabit per second throughputs. There is therefore a demand for faster and more efficient high-throughput schedulers to schedule traffic through such switches, and therefore a demand to generate computationally faster and more efficient scheduling algorithms.

30

Packet switching involves the switching of data in packets through a data network. An arriving packet could be variable or fixed length, unicast or multicast. A packet is multicast if it has more than one destination port. Variable length and/or multicast packets can be transferred to fixed-length unicast packets by methods well known in the art, and 35 the term "cell" is used to refer to a fixed-length unicast data packet. A cell consists of the

header and payload, and each cell has a unique identifier, a sequence number and the destination address (the destination output port number) of the cell which is encapsulated in the header.

5 Input Queued Switching Schemes

Figure 1 of the accompanying drawings shows a general model of an $N \times N$ switch where of the N input and N output ports, only three input and three output ports are shown for convenience and clarity. Accordingly in Figure 1, switch 1 is shown having input ports 2a, 2b, ..., 2n and output ports 3a, 3b, ..., 3n. Each input port 2a, 2b, ..., 2n is provided with one 10 (or more) input buffers 4a, 4b, ..., 4n respectively, the buffer(s) for each port being controlled by one or more buffer controllers 5a, 5b, ..., 5n respectively. In a virtual output queued input queued switch, a number of virtual output queues (VOQ) are provided: each input port having a VOQ for each destination port (i.e., each input port in the $N \times N$ switch has N VOQs) and pointers are used to point to the addresses of the cells in each VOQ. VOQs 15 are described in more detail later herein below.

A scheduler 6 determines which cells from which VOQs traverse the switch fabric 7 during a switch cycle. The function of the scheduler can be distributed between the input and output ports, such that each input and output port has an arbiter associated with it, either 20 physically or logically. Generally, a scheduler operates to switch one cell per timeslot, i.e., one cell is switched per period of time for a cell to be transmitted across the switch fabric. However, frame-based schedulers are known in the art in which a plurality of cells are switched over a plurality of timeslots. The operation of the switch is then synchronised over a plurality of fixed-size timeslots.

25

In Figure 1, the switching fabric 7 comprises a suitable interconnecting network in the form of single-stage or multiple-stage space and/or wavelength switches. Some or all of the wavelength switches can be implemented as wavelength-switched networks. The scheduler 6 is used to schedule the transmission of the cells arriving along the input links 30 8a, 8b, ..., 8n to their destination links 9a, 9b, ..., 9n.

Figure 1 for clarity only shows the possible internal input-output links 10a, 10b, ..., 10n for input port 2a, but each input port 2a, 2b, ..., 2n will have possible internal input-output links connected through the switch fabric 7 towards the appropriate destination output ports 35 3a, 3b, ..., 3n. Each internal input-output link within the switching fabric 7 is assumed to be

capable of transmitting data at a speed of one cell per timeslot. It is not necessary for an input link 8a,8b,...,8n (each of which connects to their respective input port 2a,2b,...,2n) to the switch to operate at the same speed as an internal input-output link (e.g., input-output link 10a,10b,...,10n) within the switch fabric 7.

5 During each timeslot the interconnecting network of the switch fabric 7 is capable of being configured by the scheduler to simultaneously set up a set of transmission paths between any pair of input ports 2a,2b,...,2n and output ports 3a,3b,...,3n provided no more than S cells are transmitted by an input port 2a,2b,...,2n or received by an output port 3a,3b,...,3n during each timeslot.

10

If the packet switch 1 is to process variable sized packets, or non-unicast packets, the appropriate conversion steps into fixed sized packets (or cells) is assumed to have already occurred and thus these components are not shown in Figure 1. In Figure 1, each input link 2a,2b,...,2n provides fixed sized packets to cell input buffer 4a,4b,...,4n and buffer controllers 5a,5b,...,5n respectively for header translation, addressing, and management functions which are performed on the incoming cells. The scheduler 6 processes the fixed-sized cells so that the switch fabric 7 operates in a synchronous manner. The role of the scheduler 6 is to match each cell residing in an input buffer to its destination output port. Thus the scheduler 6 can be considered to be repeatedly solving a bipartite matching problem for each timeslot, in the manner described by Anderson et al, "High-speed switch scheduling for local-area-networks", ACM Transactions on Computer Systems, vol. 11, no. 4, pp 319-352. By providing an appropriate match, e.g. matching a maximum number of input ports to output ports, or matching a maximum weighted number of input ports to output ports, in each switch cycle, the scheduler 6 is considered as treating the queued traffic in a useful and fair manner, depending on the nature of the traffic matrix.

30 In general, fixed-size packets (cells) are assumed to be switched in the switch fabric 7 to support high speed operation of the switch 1. As was mentioned above, if variable length packets are to be supported in the network, such packets are segmented and/or padded into fixed sized cells upon arrival, switched through the fabric of the switch, and reassembled into packets before departure.

35 Output contention can arise when cells destined for the same output port arrive simultaneously at the switch 1 at more than one input port. To suppress cell losses, such

cells are buffered by the switch 1 until they can be transferred to their destination output ports. The operation of the matching algorithm can potentially cause input contention, where more than one cell could be scheduled for transmission across the switch fabric from the same input port. This must be avoided by the matching algorithm. Whilst switch 1 5 supports a virtual output queuing (VOQ) scheme for the input queuing (IQ), a number of alternative queuing strategies are also known in the art, output queuing (OQ), shared queuing (SQ), and combined input-output queuing (CIOQ).

In a conventional input queued switch, basic input queuing (IQ) avoids using high- 10 bandwidth buffers by providing a buffer for each input port for incoming packets. With this queuing scheme, the bandwidth demand of each input buffer is reduced to at least one write operation and one read operation per time slot. With a properly designed scheduling algorithm, a set of input-output contention free cells is selected from the buffered cells for transmission to their destination output ports, from time slot to time slot. When the overall 15 scheduling operation is applied to a number of timeslots simultaneously in a frame of timeslots, scheduling comprises two sub-processes, matching and timeslot assignment. These processes will be described in more detail later herein below.

Whilst output queued (OQ) switches and shared queue (SQ) switches can generally 20 achieve better performance than input queued switches and combined input-output queued switches, this is only so for a finite size of $N \times N$ switch. As the number of input and output ports of the switch increases, the bandwidth demand of the OQ or SQ buffer grows linearly as the aggregated input-output link rate increases. Accordingly, it is known in the art that OQ and SQ switches generally do not scale very well. As the switch 25 architecture of an input queued (IQ) switch with FIFO queuing (and similarly a Combined Input-Output Queued (CIOQ) switch) is much simpler than that of OQ and SQ switches, IQ and CIOQ switches generally scale better than OQ and SQ switches as each input buffer maintains a single FIFO for all incoming cells. However, despite the simplicity in the switch architecture of an IQ switch with FIFO queuing, the maximum throughput is 30 only 58.2% for uncorrelated (Bernoulli) traffic with destination outputs distributed uniformly, and the throughput is worse for correlated (on/off bursty) traffic. This is a result of the HOL blocking problem, in which a cell queuing behind the HOL cell of a FIFO cannot participate in scheduling, even if both its residing input and destination output are idle.

By supporting Virtual Output Queuing (VOQ) in the input ports of an IQ switch, HOL blocking can be removed. The Virtual Output Queue (VOQ) scheme (also known as the multiple input queuing scheme) is described in "The iSLIP Scheduling Algorithm for Input-Queued Switches" by N. McKeown, IEEE/ACM Trans. Networking, Vol. 7, No. 2, pp. 188-5 200 (April 1999), and United States Patent Number US 5500858, the contents of which are hereby incorporated by reference).

Conventionally, in an input-buffered VOQ switch, a fixed-size packet (or cell) is sent from any input to any output, provided that, in a given timeslot, no more than one cell is sent 10 from the same input, and no more than one cell is received by the same output. Each input port has N VOQs, one for each of N output ports. The HOL cell in each VOQ can be selected for transmission across the switch in each timeslot. Accordingly, in each timeslot, a scheduler has to determine one set of matching, i.e., for each of the output ports, the scheduler has to match one of the corresponding VOQs with the output port.

15 Figure 2 of the accompanying drawing shows schematically a 4 x 4 VOQ IQ switch 20. Switch 20 has four input ports #a1, #a2, #a3, and #a4 and four output ports #b1, #b2, #b3, and #b4 which are capable of being interconnected by an internal switch fabric 21. Each input port #a1, #a2, #a3, and #a4 has four VOQs, one VOQ for each of the destination 20 output ports #b1, #b2, #b3, and #b4. In Figure 2, the VOQs are denoted VOQ#ai#bj where i, j ranges from 1 to 4 respectively.

It is known in the art that the implementation of a VOQ scheme can enable up to 100% throughput to be achieved. Scheduling algorithms such as maximum weight matching 25 algorithms have a high level of complexity, e.g. the number of calculations to be performed per single time-slot matching is $O(N^3)$. Currently, the amount of time it would take to perform such an algorithm to calculate the matching is impractical under high-speed environments where the duration of a time slot (the time taken to run a switching cycle i.e., to transport a cell from an input port to its destination port across the switch 30 fabric) is very small.

Other scheduling schemes known in the art include a three-stage switch scheduling scenario in which a large packet switch is decomposed into a number of smaller switches having fewer ports (see Joseph Y Hui in "Switching and Traffic Theory for Integrated 35 Broadband Networks", Kluwer Academic Publishers, 1990, Chapt. 5, and J S Turner,

"WDM Burst Switching for Petabit Data Networks", OFC 2000 presentation). However, in these known schemes each switch has its own buffering which requires scheduling to be performed independently for each switch. No scheduling occurs between the switches, which leads to at least two limitations. Firstly, heuristic rules are required (e.g. load spreading between switches) to enable a reasonable switch performance to be maintained. Secondly, packets may arrive at their destination out of sequence.

Contention Resolution in an Input Queued Virtual Output Queued Switch

Consider the $N \times N$ switch shown in Figure 1. In Figure 1, the switch 1 has N input 10 queues in each input port. Accordingly, there are N^2 VOQs in total. However, switch 1 has only N output ports to transfer at most N cells to in a given timeslot. Thus contention occurs amongst the N^2 VOQs.

Several methods are known in the art which seek to resolve this contention issue. One 15 known technique to reduce the computing time complexity is to use a heuristic maximal-sized matching algorithm such as the iSLIP scheduling algorithm by N.McKeown. Like many matching algorithms, iSLIP comprises 3 phases known as the request, grant, and accept phases. In the request phase, each of the N^2 input queues sends a request to the output ports. In the grant phase, each of the output ports grants one request among its 20 own receiving requests using a suitable selection operation and notifies the result of grant to each of the input ports. An input port may receive several grants from each output port at the same time so that in the accept phase each of the input ports accepts one grant amongst its own receiving grants using a suitable selection process. Several request-grant-accept cycles are iteratively performed.

25 With such a three-phase matching approach, a problem which needs to be addressed to optimise the matching process is how to ensure that the selection processes fairly and quickly select and grant one request from a plurality of requests which could be granted (and equivalently accept one grant from a plurality of grants which could be accepted). In 30 iSLIP, this selection process is achieved using a particular set of pointer rules. iSLIP can be faster than alternative schemes which use random selection. Unfortunately, in the iSLIP algorithm, as the number of input ports and output ports increases, the number of requests and grants which must be selected between in the grant and accept phases within one time slot increases. Although the iSLIP algorithm has less computing 35 complexity than a maximum matching algorithm it has a limitation: iSLIP requires the

maximal matching to be completed within one timeslot. Again, as the switch size increases or if a switch has very high port speeds (either because the matching time itself increases beyond the time for one time slot, or because the timeslot itself has a shorter duration) iSLIP is no longer suitable.

5

Several other matching schemes have been devised in the art which seek to provide greater scalability and so support faster switch cycles. For example, pipeline-based scheduling algorithms such as the Round-Robin Greedy Scheduling (RRGS) allows each input to perform only one round-robin arbitration within a given time slot to select one

10 VOQ. For a switch with N inputs, N input round-robin operations (to select a cell to be transmitted at a given time slot T) are allocated to the different previous N time slots $\{T-N, T-(N+1), \dots, T-1\}$ in a simple cyclic manner to avoid output contention. A drawback of this scheme is when traffic is not balanced across the input of the switch, some inputs can unfairly send more cells than others. Whilst other schemes are known in the art to
15 guarantee pre-reserved bandwidth, for example, the weighted RRGS scheme, this has a drawback in that it does not guarantee fairness for best-effort traffic and a further draw back in that as every even number of timeslot cycles an idle timeslot is produced resulting in the switch capacity not being fully used.

20 Overview of Frame-Based Scheduling

As was discussed briefly in the introduction, the overall scheduling operation comprises two sub-processes, a matching process and a timeslot assignment process. A similar division exists where a frame-based scheduling approach is implemented.

25 The frame based approach comprises two steps for each frame. The first step involves a matching process in which a number of cells queued at the inputs are accepted for transmission to outputs in a non-contentious manner. The second step involves a time-slot assignment process in which the successfully matched cells are scheduled for transmission in the different time slots of the frame. This time-slot assignment step can
30 be considered to be equivalent to scheduling a set of non-conflicting requests in a time-frame, which can be performed using known path-searching algorithms such as those used to route circuits in a Clos interconnection network, for example, see WO01/67783 "Switching Control" and also WO01/67803 "Frame Based Algorithms for Switch Control", and WO01/67802 "Packet Switching", all three of which are hereby incorporated by
35 reference.

At the beginning of a frame, the total number of packet requests from each input port to each output port as a pair is collected into an $N \times N$ Request Matrix R (the request phase of the process). Each element $r(i,j)$ of this matrix is an integer showing the total amount of stored packets in the VOQ between input port i and output port j .

The matching process populates a symmetric $N \times N$ Accepted-Requests matrix A . Each element $a(i,j)$ of A represents the total number of switching requests from the VOQ between input port i and output port j that have been accepted to be switched during the following time period (frame) available for transferring one or more cells between an input port and an output port over using one or more timeslots. Each $a(i,j)$ of A is constrained by the overall capacity of the switch input and output ports "F", i.e., the sum of elements in each row and each column must not exceed the frame size F ; i.e. the number of time slots or cells in the frame. Various matching algorithms are known in the art to try to optimise the use of the available switch capacity. All of these matching algorithms seek in each time period consisting of one or more time slots, a non-conflicting match to be determined between the input ports and the output ports of a switch fabric of an $N \times N$ symmetric request matrix.

For example, where $F = 1$ (and for unicast traffic) a matching process will seek to link each input port to at most one output port and each output port is linked to at most one input port. A complete matching of the input ports to the output ports in one timeslot is then equivalent to determining the appropriate permutation of the input ports. However, as a complete matching cannot always be achieved, maximal matching algorithms seek to optimise the selection of which cells should be transmitted from input to output per timeslot. This optimisation depends on a number of factors selected according to the particular embodiment of the matching algorithm implemented and can depend, for example, on the length of queue and/or how long the cell at the head of each queue has been queued for.

30

Frame-based matching where $F \geq 1$ has already been described in the art, for example, in "Frame-based matching algorithms for input-queued switches" by Andrea Bianco, Mirko Franceschinis, Stefano Ghisolfi, Alan Michael Hill, Emilio Leonardi, Fabio Neri, Rod Webb, HPSR 2002, Workshop on High Performance Switching and Routing, Kobe, Japan, 26-29 May 2002, the text of which is incorporated herein by reference. Bianco et al describe a

frame-based switch contention resolution scheme which can be considered to comprise two-steps for each frame (a frame being considered to be a set of one or more time-slots).

Consider a frame whose length is F (i.e., whose transmission could occupy F consecutive timeslots). A set of cells selected to be transmitted in the timeslots belonging to the next frame is selected at the end of the current frame, i.e., switch control occurs on a multi-time-slot basis at the edge of each frame boundary. The set of cells selected for transmission is termed the F -match, and this needs to always comply with the joint criteria that i) the total number of selected cells from each input port cannot be larger than F and ii) the total number of selected cells which are to be transmitted to each output port cannot exceed F . Equivalently, if $a_{i,j}$ is the number of accepted cells from input i to output j , the constraints are:

Eqn. 1 $\sum_i a_{i,j} \leq F \quad \forall j$ and

Eqn. 2 $\sum_j a_{i,j} \leq F \quad \forall i$.

The selection of the cells forming the set to be transmitted in the next frame is made using an F -matching algorithm (and where $F=1$, the F -matching is equivalent to a conventional time-slot by time-slot approach). Once the F -match has been obtained, cell transmissions need to be assigned to different timeslots of the frame, i.e., a set of at least F switch permutations capable of transferring all cells belonging to the F -match in a non-conflicting manner.

The frame-based matching scheme Bianco et al describe is implemented using a request/grant/accept scheme in a manner similar to iSLIP. iSLIP utilises a rotating priority scheme in which the selection of requests to be granted (at outputs) and of grants to be accepted (at inputs) is implemented using two sets of N pointers, one for each input and one for each output. An output (input) pointer points to the input (output) port to which highest priority is given in issuing grants (acceptance). Accordingly, grants and acceptances are given to the first busy queue in a cyclic order starting from the current pointer position. Input and output pointers are up-dated after each matching to the first input (output) following the one which has been accepted.

Bianco et al also describe a "No Over Booking" (NOB) matching algorithm consisting of a generalisation of the known iSLIP algorithm by McKeown et al, but one or more iterations (i.e., a generalisation of 1-SLIP) and associated pointer update rules. The NOB algorithm output booking and input booking steps are described in detail in Bianco et al, and are

5 incorporated herein by reference. Briefly, the NOB algorithm steps through an output booking phase followed by an input booking phase, similarly to iSLIP. In the output booking phase, each virtual output queue (VOQ) requests a number of time-slots in the appropriate output frame, and as a reply each output port issues up to F grants distributed amongst the N VOQs destined for that output. The total number of requests is

10 represented by a request matrix R , whose elements $r_{i,j}$ represent the total number of time slots requested by input port i for output port j .

In general the length $q_{i,j}$ of each VOQ (i.e. the number of cells queued) can be greater than the frame length F . The number of actual requests made by each VOQ from the

15 request matrix R is up to $q_{i,j}$ when $q_{i,j} < F$ but if $q_{i,j} \geq F$ then up to F (as no more than F time slots can be requested at any one time). The request matrix R is distinguished from the normalisation phase matrix to be discussed below which uses as its input "requests" the actual queue lengths, but which does not reduce the number of requests in each VOQ to the frame length F prior to the first stage of matching.

20 During the output booking phase, each output port operates simultaneously, hence output ports operate independently, so that there is no guarantee that the total number of grants received by VOQs at one input port will not exceed the capacity of the input frame. To remedy this, each input port accepts up to F of the grants received at that

25 port. Each acceptance received by a VOQ at one input port gives that port the right to transmit one cell in the next frame.

The NOB frame matching algorithm Bianco et al describe is in some sense therefore a

30 hybrid between a maximum weight matching (MWM) (which assigns a weight to the cells at the head of each VOQ, and which optimises the cumulative weight of cells which are successfully matched) and a maximum size matching (MSM) (which addresses optimising on the basis of the overall number of cells which are successfully matched being a maximum).

In each phase, the final steps in the above algorithms begin on an initial VOQ which is indicated by a pointer. Accordingly, each output port maintains a pointer showing which input port should be given priority for its additional grants in the final output booking step,

5 and each input port keeps a pointer showing which output port has priority in its final input booking step. Several schemes are known in the art for updating the pointers so that a level of fairness is maintained.

Prior art such as United States Patent number US 6,487,213 entitled "Methods and

10 Apparatus for Fairly Arbitrating Contention for an Output Port" by Chao describe hierarchical arbitration methods in which requests are aggregated together, but arbitration is performed independently for different output ports. Matching is not performed globally between all input and output ports to solve contention across the entire switch fabric, nor does Chao address the issue of resolving contention in an input-switch, as Chao

15 addresses the issue where both input and output queuing are provided.

Consider when the original request matrix R_0 is transformed to a normalised request matrix R_{norm} by transforming factor d . The original request matrix R_0 could be, for example, the matrix of VOQ queue lengths (i.e. numbers of requests or cells queued in each VOQ)

20 or a measure of the requested traffic rates. An example of a transforming factor d is given below.

$R_r = R_0 - R_{norm}$ is then the request matrix of remaining requests given by the original request matrix R_0 and R_{norm} is the partially populated Accepted-Requests matrix A from

25 the first stage of the matching. R_r is used to fill up as much of the remaining capacity of the frame as possible, by running another matching algorithm (which could be the same as the first or different) in a second stage to populate a second accepted requests matrix A_2 derived from the matrix of remaining requests R_r . The final matrix of accepted requests $A = R_{norm} + A_2$, i.e., A is just the sum of the two matrices found during the two

30 stages. Referring back to the example given above with reference to Figure 2 of the accompanying drawings, the request matrix

$$\text{Eqn. 3} \quad R = \begin{pmatrix} 3 & 4 & 2 & 0 \\ 5 & 0 & 1 & 0 \\ 8 & 5 & 1 & 3 \\ 2 & 0 & 2 & 6 \end{pmatrix}$$

is transformed by a transformation factor $d = F/\max(F, mval)$, where $F = 8$ and $mval$ is defined to be the maximum sum of any one column or row in R , i.e.,

$$\text{Eqn. 4} \quad mval = \max \left(\sum_{i=1}^4 a_{i,j} \leq F \quad \forall \quad j, \quad \sum_{j=1}^4 a_{i,j} \leq F \quad \forall \quad i \right).$$

Here $d = 8/18$ and thus $R_0 = \lfloor 4R/9 \rfloor$, where the elements of R_0 are the integer parts of the resulting numbers.

For this example, then

$$\text{Eqn. 5} \quad R_{\text{norm}} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 3 & 2 & 0 & 1 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

The remaining request matrix =

$$\text{Eqn. 6} \quad R_r = R_0 - R_{\text{norm}} = \begin{pmatrix} 2 & 3 & 2 & 0 \\ 3 & 0 & 1 & 0 \\ 5 & 3 & 1 & 2 \\ 2 & 0 & 2 & 4 \end{pmatrix}$$

15 A second matching procedure is then performed on the remaining request matrix R_r which produces another Accepted Requests matrix A_2 . The total Accepted Requests Matrix A per frame is then given by the sum of R_{norm} and A_2 , i.e., $A = R_{\text{norm}} + A_2$.

20 Therefore the operation of an existing, example single-level matching algorithm known in the art can be summarised as follows.

Normalisation Stage

First the elements in the request matrix are transformed by normalising them according to the highest queue value and the total number of timeslots available in a frame, so that in

5 the normalisation phase:

$$\text{Eqn. 7 } [r(i,j)] \Rightarrow [r_{\text{norm}}(i,j)], r_{\text{norm}}(i,j)$$

"No Overbooking" Stage

10 This comprises an output booking phase followed by an input booking phase. In the output booking phase, a granted-request matrix is formed from the matrix of remaining requests, i.e., in the output port booking phase the grants are derived as follows:

$$\text{Eqn. 8 } [r_{\text{norm}}(i,j)] \Rightarrow [g(i,j)] ; \quad \sum_i g(i,j) \leq F - [\sum_i Q_{\text{norm}}(i,j)],$$

where Q_{norm} is the number of normalised requests queued for a particular output port. In the input booking phase, an accepted grant matrix is generated from the matrix of granted requests, i.e., the accepted grants populate this matrix according to:

$$\text{Eqn. 9 } [g(i,j)] \Rightarrow [a(i,j)] ; \quad \sum_j a(i,j) \leq F$$

20

Time Slot Assignment

The second process of the scheduling algorithm is the Time Slot Assignment. It attempts to compute the set of switch (or network) configurations for each time slot, such that the matrix of accepted requests can be transferred from the input ports to the output ports

25 across the switch without blocking any packet, i.e., to ensure there is a free time slot available for each packet from its input port to its desired output port. This is not always possible, depending on the Time Slot Assignment algorithm and the number of time slots (switch permutations) available. Some or even all of this set of switch permutations may be the same. As an example, consider the request acceptance matrix

$$\text{Eqn. 10 } A = [a(i,j), 1 \leq (i,j) \leq 4] = \begin{bmatrix} 2 & 4 & 2 & 0 \\ 3 & 1 & 3 & 1 \\ 2 & 3 & 1 & 2 \\ 1 & 0 & 2 & 5 \end{bmatrix}$$

A possible set of 8 switch permutations to send these numbers of cells or packets from input ports to the output ports (elements in the table) across the switch is shown in the following table

Time slot ►	1	2	3	4	5	6	7	8
Input Port ▼								
1	1	1	2	2	2	2	3	3
2	3	4	1	3	1	3	2	1
3	2	2	4	4	3	1	1	2
4	4	3	3	1	4	4	4	4

5

If we call this set of permutations P_n , where n is the time slot within the frame ($1 \leq n \leq F$), then it would correspond to the following sequence of permutation matrices

$$P_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad P_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \dots \quad P_8 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

10 Eqn. 11

Several algorithms are known in the art suitable for implementing time slot assignment, with different blocking characteristics, dependent also on the number of time slots (switch permutations) available.

15

Referring now to Figure 2 of the accompanying drawings, an input queued cell (or packet) switch arrangement is shown. In Figure 2, each of the four input ports #a1..#a4 has a first-in-first-out virtual output queue VOQ, designated as VOQa1b1...VOQa4b4. Each 20 VOQ queue stores cells destined for a different output port, the total length of the queue from input port i to output port j (the number of cells to be transmitted in the next time frame) being indicated by $q_{i,j}$ where $i, j = 1, 2, \dots, N$. Cells which queue up in the VOQs generate requests which can be presented by a queue matrix as shown below (and in Figure 2). In general, the number of cells queued in a VOQ can exceed the frame size F .

25

As an example of a conventional single-level matching an example queue matrix $[Q(i,j)]$ for a 4x4 switch having a frame duration F of F=4 time slots is matched below. The VOQ lengths, i.e. the number of cells or packets waiting in each VOQ, are assumed to have a "powers of two" distribution, i.e.

5 Eqn. 12

$$[Q(i,j)] = \begin{bmatrix} 1 & 2 & 4 & 8 \\ 2 & 4 & 8 & 1 \\ 4 & 8 & 1 & 2 \\ 8 & 1 & 2 & 4 \end{bmatrix}$$

A conventional single-level frame-based matching process in which a number of stages are present in the matching process will now be described.

10 Conventional Single-Level Matching Normalisation Stage

The normalisation algorithm first finds the row (input port) or column (output port) with the largest sum of requests (or queue lengths), maxval. Every request (or queue length) is then normalised to this maximum value, firstly by being multiplied by the ratio $c=F/\text{maxval}$ if maxval is $>F$ (or greater than the maximum number of grants or acceptances allowed)

15 and $c=1$ otherwise, and secondly by taking the integer part of the resulting number.

For $[Q(i,j)]$ in Eqn.12 maxval = 15, which is larger than F (=4). Hence the normalised queue matrix becomes

Eqn. 13

$$[Q_{\text{norm}}(i,j)] = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 0 \\ 1 & 2 & 0 & 0 \\ 2 & 0 & 0 & 1 \end{bmatrix}$$

20

All of these cells or packets are assumed already to be granted by the output ports and accepted by the input ports. The request matrix presented to the second "no overbooking" stage is the difference between the original queue matrix and the normalised queue matrix, i.e. the remaining requests

25

$$\text{Eqn. 14. } [r(i,j)]_{\text{output}} = [Q(i,j)] - [Q_{\text{norm}}(i,j)]$$

$$= \begin{bmatrix} 1 & 2 & 4 & 8 \\ 2 & 4 & 8 & 1 \\ 4 & 8 & 1 & 2 \\ 8 & 1 & 2 & 4 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 0 \\ 1 & 2 & 0 & 0 \\ 2 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 2 & 3 & 6 \\ 2 & 3 & 6 & 1 \\ 3 & 6 & 1 & 2 \\ 6 & 1 & 2 & 3 \end{bmatrix}$$

These requests are used by the "no overbooking" stage to fill up the remaining available time slots in the frame as much as possible.

5

Conventional Single-Level Matching "No Overbooking" Stage Output Booking Phase

The number of requests in effect already granted by the output ports in the normalisation stage is

Eqn. 15

$$[\sum_i Q_{\text{norm}}(i,j)] = [3 \ 3 \ 3 \ 3]$$

10

The remaining number of grants available in each output port is therefore

Eqn. 16

$$[F \ F \ F \ F] - [3 \ 3 \ 3 \ 3] = [1 \ 1 \ 1 \ 1]$$

Output booking operates simultaneously in each output port in the three following steps:

15

1. if the total number of requests received by the port is less than the remaining number of grants available, then all requests to the port are granted.
2. if the number of VOQs with an unsatisfied request destined for the port is less than or equal to the remaining number of grants, then the VOQs receive one grant each. This step is repeated as many times as possible.
3. taking the VOQs in turn, starting from the one indicated by a pointer, each VOQ with an unsatisfied request receives one grant until the total number of grants given by the port reaches the remaining number of grants in Eqn.16.

25 Step 1 does not apply in this case, because the total number of requests $[\sum_i r(i,j)]_{\text{output}}$ is 12 to all ports (from Eqn.14). Step 2 does not apply either, because there are 4 VOQs with unsatisfied requests destined for every output port and only one available grant each. Step 3 applies in this case.

30 Conventional Single-Level Matching Pointer Update Rules

A deterministic NOB25 pointer up-date rule such as that described by Bianco et al initialises the pointers so that each output port gives priority to a different input port (and

vice versa) and the pointer advances by 1 each frame, such that on cycle k of the algorithm (i.e. for the k -th frame, starting at $k=0$), port P gives priority to port p , where

Eqn. 17
$$p = 1 + [(LN - P + k)_{\text{mod } LN}]$$

5

Hence in the first frame $k=0$, with $LN=4$ in this example, output port 1 points to input port 4, 2 points to 3, 3 points to 2 and 4 points to 1, i.e. the pointers point to VOQ requests $r(4,1)$, $r(3,2)$, $r(2,3)$ and $r(1,4)$ in Eqn.14. All of these VOQs have 6 requests, and because Eqn.16 allows only 1 more available grant for each output port, each of these four VOQs 10 will be granted one more request, i.e. the additional output booking grants $[g(i,j)]$ are given by

Eqn. 18
$$[g(i,j)] = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

15 **Conventional Single-Level Matching "No Overbooking" Stage Input Booking Phase**

The number of requests in effect already accepted by the input ports in the normalisation phase is

Eqn. 19
$$[\sum_j Q_{\text{norm}}(i,j)] = \begin{bmatrix} 3 \\ 3 \\ 3 \\ 3 \end{bmatrix}$$

20 The remaining number of additional acceptances available in each input port is therefore

Eqn. 20
$$\begin{bmatrix} F \\ F \\ F \\ F \end{bmatrix} - [\sum_j Q_{\text{norm}}(i,j)] = \begin{bmatrix} 4 \\ 4 \\ 4 \\ 4 \end{bmatrix} - \begin{bmatrix} 3 \\ 3 \\ 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

The request matrix for this input booking phase is the additional output booking grants matrix $[g(i,j)]$ (Eqn.18). Step 2 of the "no overbooking" algorithm applies, so all of the 25 additional output booking grants are accepted, i.e. the additional input booking acceptances $[a_{\text{additional}}(i,j)]$ are given by

Eqn. 21
$$[a_{\text{additional}}(i,j)] = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

The final acceptance matrix is the sum of the acceptances from the initial normalisation (Eqn.13) plus these additional acceptances from the "no overbooking" algorithm (Eqn.21), i.e.

Eqn. 22 $[a(i,j)] = [Q_{\text{norm}}(i,j)] + [a_{\text{additional}}(i,j)]$

$$\begin{aligned}
 5 \quad &= \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 0 \\ 1 & 2 & 0 & 0 \\ 2 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \\
 &= \begin{bmatrix} 0 & 0 & 1 & 3 \\ 0 & 1 & 3 & 0 \\ 1 & 3 & 0 & 0 \\ 3 & 0 & 0 & 1 \end{bmatrix}
 \end{aligned}$$

All input and output ports fill all $F=4$ time slots in this first frame. A full set of 16 cell or packet requests has been accepted in the first cycle or frame. They are taken from 8 of 10 the longest VOQs in Eqn.12.

A matching algorithm may not completely fill up the matrix of Accepted-Requests from the matrix R (i.e., ensure $A - R = \text{a null matrix}$, indicating all requests have been granted). This is because A is constrained by the fact that the sum of elements $a(i,j)$ in each row and 15 column cannot exceed the number of timeslots in a frame length, F . Consider where $F =$

8 timeslots, then both $\sum_{i=1}^4 a_{i,j} \leq 8 \quad \forall j$ and $\sum_{j=1}^4 a_{i,j} \leq 8 \quad \forall i$ and by the constraint

that the total of the sums of elements in each column and each row is limited to the total number of available timeslots $N \times F$, i.e.,

$$\sum_{i=1}^4 \sum_{j=1}^4 a_{i,j} \leq N \times F,$$

20 where $N = 4$ and $F = 8$ in this example.

Thus, as a more specific example, consider where the matching algorithm populates a matrix of Accepted-Requests as follows:

$$A = \begin{pmatrix} 2 & 4 & 2 & 0 \\ 3 & 0 & 1 & 0 \\ 2 & 3 & 1 & 2 \\ 1 & 0 & 2 & 5 \end{pmatrix}$$

Here $F = 8$ and $N \times F = 32$, however, $\sum_{i=1}^4 \sum_{j=1}^4 a_{i,j} = 28$ meaning that the maximum switch capacity of 32 requests per time period of 8 time slots has not been utilised.

The above example illustrates clearly one limitation of such a known frame-based

5 matching process, in that the switch capacity may not be utilised fully, resulting in some redundant switch capacity in any given frame.

Summary of the Invention

The present invention seeks to obviate and/or mitigate some of the problems related to

10 optimising matching algorithms so that their computational complexity is further reduced, yet which can more efficiently utilise the switch capacity. Ideally the computational complexity is reduced to a level which is suitable for the high-speed switches which are currently being developed for future use. The invention provides a frame based matching algorithm which seeks to obviate and/or mitigate some of the problems known in the art

15 related to optimising matching algorithms by seeking to further reduce the number of computing steps in a frame-based matching process from $O(LN)$ to $O(L)$ or $O(N)$ for the frame.

A first aspect of the invention provides a matching method for a number N of first

20 elements, each first element arranged to at least provide ingress to a switch arrangement, each of the first N elements comprising a number L_1 of first subelements, the switch arrangement having a number ML_2 of second subelements arranged to at least provide egress from said switch arrangement, and wherein each of the first L_1 subelements is capable of conveying a service request for at least one of said second subelements ML_2 ,

25 wherein the method comprises: firstly, for every one of the N first elements, aggregating service requests from all L_1 first subelements to each of the ML_2 second subelements, and secondly, resolving contention for said service requests from all N first elements to one or more of said second ML_2 subelements, and thirdly, for each first element, resolving contention between the L_1 subelements and said second ML_2 subelements.

30

The step of resolving contention between the L_1 subelements and said second ML_2 subelements may be performed in parallel for each said first element.

The ML_2 second subelements of the switch arrangement may be provided as a number M of second elements, each of said M second elements being associated with a number L_2 of second subelements.

- 5 Each subelement may be capable of generating at least one said service request.

The first subelements and said second subelements may be bi-directional and provide both ingress and egress from the switch fabric. The first subelements may comprise said second subelements.

- 10 The first subelements and said second subelements may be unidirectional and then said first subelements may provide ingress and said second subelements may provide egress from the switch arrangement.

The first and second subelements may comprise ports in the switch arrangement and said

- 15 first elements comprise aggregations of said first subelements.

The first and second subelements may comprise ports in the switch arrangement, and the first elements may comprise aggregations of said first subelements and said second elements comprise aggregations of said second subelements.

20

The switch arrangement may comprise an input queued cell switch and said service requests comprise requests for transmitting a service information rate from one of said first subelements to at least one of said second subelements.

- 25 The switch arrangement may comprise an input queued cell switch and said service requests comprise requests for transmitting at least one cell from one of said first subelements to at least one of said second subelements.

- 30 The switch arrangement may comprise an input queued packet switch and said service requests comprise requests for transmitting a service information rate from one of said first subelements to at least one of said second subelements.

- 35 The switch arrangement may comprise an input queued packet switch and said service requests comprise requests for transmitting at least one packet from one of said first subelements to at least one of said second subelements.

The packets may have a fixed-length and comprise cells and said packet switch may be an input queued cell switch arranged to switch fixed-length cells, and said service requests may comprise requests for transmitting one or more fixed-size cells from one of

5 said first subelements to one or more of said second subelements.

The packets may have a fixed-length and comprise cells and said packet switch may be an input queued cell switch arranged to switch fixed-length cells, and said service requests may comprise requests for transmitting a service information rate from one of

10 said first subelements L_1 to one or more of said second subelements L_2 .

The switch arrangement may comprise a circuit based switch and said service request comprises a request for a connection in a circuit-based switch.

15 The switch arrangement may comprise a circuit based switch and said service request comprises a request for a bandwidth in a circuit-based switch.

The switch arrangement may comprise a circuit based switch and said service request comprises a request for a service information rate in a circuit-based switch.

20

The service information rate may be a bit rate.

The circuit based switch arrangement may comprise at least one switch taken from the group consisting of:

25 any known time-domain, frequency domain, wavelength domain or space domain switching technologies.

The circuit-based switch arrangement may comprise a combination of said switches.

30

The switch arrangement may comprise a network, and said elements may comprise aggregations of network terminals or nodes and said subelements may comprise network terminals or nodes.

The switch arrangement may comprise an arrangement of inter-connectable sub-networks, where said elements comprise sub-networks and said subelements comprise network terminals or nodes.

5 The network may be an optical network.

The sub-networks may comprise optical networks.

10 The elements may become subelements with respect to elements in a higher layer of matching.

Multiple layers of matching may be performed in a hierarchy of matching levels.

15 A second aspect of the invention provides a method as claimed in any previous claim, wherein the method of matching comprises: firstly, aggregating service requests to the highest level of the matching hierarchy, and secondly, resolving contention for said service requests at the highest level of the matching hierarchy, and thirdly, resolving contention in turn down through the matching levels to the lowest level of matching.

20 A third aspect of the invention seeks to provide a matching method for a switch arrangement comprising a plurality N of input elements, each input element comprising a plurality (L_1) of input subelements, and a plurality M of output elements, each output element comprising a plurality L_2 of output subelements, the matching method comprising the following steps: performing a first matching across the switch fabric for each of the 25 plurality of N input elements and the L_2 subelements by performing the steps of: summing a number of requests from each of the L_1 subelements of the input element; generating a first $N \times ML_2$ request matrix; matching the first request matrix to generate a first matrix of accepted requests; and performing a second matching across the switch fabric for each of the N input elements by performing the steps of: generating N asymmetric second $L_1 \times 30 ML_2$ matrices, for each of the N input elements; and matching each of the N asymmetric second matrices to generate N second matrices of accepted requests; and generating a $NL_1 \times ML_2$ matrix of accepted requests from the first $N \times ML_2$ matrix of accepted requests and the N second $L_1 \times ML_2$ accepted request matrices.

35 The $NL_1 \times ML_2$ matrix of requests may be symmetric.

L_1 may be equal to L_2 and N may be equal to M .

The subelements may comprise nodes in an optical ring network.

5

The subelements may comprise terminals in a passive optical network.

The switch arrangement may comprise a packet switch arrangement.

10 The packet switch arrangement may be capable of switching fixed-length packets.

The switch arrangement may comprise a cell switching arrangement.

The cell switching arrangement may be capable of switching packets.

15

A fourth aspect of the invention seeks to provide a switch arrangement, the switch arrangement having number N of first elements, each first element arranged to at least provide ingress to a switch arrangement, each of the first N elements comprising a number L_1 of first subelements, the switch arrangement having a number ML_2 of second

20 subelements arranged to at least provide egress from said switch arrangement, and wherein each of the first L_1 subelements is capable of conveying a service request for at least one of said second subelements ML_2 , wherein said service requests are conveyed by performing a matching method which comprises:

firstly, for every one of the N first elements, aggregating service requests from all

25 L_1 first subelements to each of the ML_2 second subelements, and

secondly, resolving contention for said service requests from all N first elements to one or more of said second ML_2 subelements, and

thirdly, for each first element, resolving contention between the L_1 subelements and said second ML_2 subelements.

30

In the fourth aspect, the matching method may be according to any one of the first, second or third aspects.

A fifth aspect of the invention seeks to provide a network including a switch arrangement

35 according to the fourth aspect.

A sixth aspect of the invention seeks to provide a suite of at least one computer programs arranged when executed to implement steps in a method according to the first, second or third aspects.

5

At least one program may be arranged to be implemented by software running on a suitable computational device.

At least one program may be arranged to be implemented by suitably configured
10 hardware.

A sixth aspect of the invention seeks to provide a scheduler for a switching arrangement, the scheduler arranged to perform a scheduling process, the scheduling process comprising:

15 a matching method according to any one of the first, second or third aspects; and
a time-slot assignment process.

A seventh aspect of the invention seeks to provide a matching method according to any one of the first, second or third aspects wherein the subelements comprise ports, and the
20 matching updates the pointers to input ports according to the following rule: $p_{out} = 1 + [(LN - P_{in} + k)_{mod LN}]$ and the output ports are updated according to the following rule: $p_{in} = 1 + [(LN - P_{out} + k)_{mod L}]$

An eighth aspect of the invention seeks to provide a matching method according to any
25 one of the first, second or third aspects wherein the subelements comprise ports, and the matching updates the pointers to input ports according to the following rule: $p_{out} = 1 + [(LN - P_{in} + k)_{mod LN}]$ and the output ports are updated according to the following rule: $p_{in} = 1 + [(m - P_{out} + k)_{mod L}]$

30 Advantageously, the invention seeks to provide a scheduling algorithm suitable for a high-performance VOQ IQ switch which has a reduced level of complexity yet supports an acceptable level of throughput. The scheduling algorithm is provided with less computational complexity by performing the scheduling over several hierarchical levels within and between smaller switches or sub-networks by providing a matching algorithm
35 which operates on an asymmetric request matrix.

Advantageously, the invention reduces the computing complexity and enables larger cell/packet switches/networks to be constructed without distributing the scheduling decisions too loosely between the smaller switches or sub-networks so that performance 5 is degraded.

The asymmetric request matrix grants requests between inputs and outputs of differing levels of aggregation, e.g., switch-port, ring-node, or PON-terminal. In general, as more than one hierarchical level of matching is performed between different sub-networks, 10 multistage buffering and switching can be used to support this. Advantageously, however, in some embodiments of the invention, the multi-stage buffering/switching is implemented by means of multi-hopping. Advantageously in such embodiments, buffering remains at the switch/network edge. This means that where the invention is implemented in an otherwise optical network environment, the buffering can be implemented electronically, 15 avoiding the expense of optical buffering technology.

Advantageously, the invention can provide a global frame-based optimal scheduling algorithm which operates both within and between each individual sub-switch/network, the scheduling algorithm comprising a matching algorithm stage and a channel assignment 20 (time-slot assignment) stage. The global frame-based multi-level matching scheme uses multiple aggregation levels. Channel assignment can be provided by any suitable mechanism, for example, as provided by the inventor's co-pending patent application entitled "TIME-SLOT ASSIGNMENT PROCESS" co-filed herewith, the contents of which are incorporated by reference. In a preferred embodiment of the invention the channel 25 assignment stage comprises a method of buffering the timeslot interchanging stages by multi-hopping (3 hops) between sub-sets of the network nodes (terminals) so that buffering can be located at the edge nodes only as described by said co-pending patent application entitled "TIME-SLOT ASSIGNMENT PROCESS".

30 In contrast, to the prior art, the invention uses asymmetric traffic request matrices, applied to different parts of the overall network with different levels of aggregation, in order to reduce the matching complexity. For example, the asymmetric request matrix can be between input switch-output port or upstream ring-downstream node or upstream PON-downstream terminal. This allows sufficient information about individual port, node or

terminal identities to be retained to prevent receiver contentions and source blocking, while reducing the overall matching complexity.

The invention will now be described with reference to the following drawings which are by 5 way of example only and in which:

Figure 1 is a schematic sketch of a $N \times N$ input queued packet switch;

Figure 2 is a schematic sketch of a 4×4 input queued packet switch showing virtual 10 output queues VOQs and its corresponding request matrix;

Figure 3a is a sketch showing the input and output elements and subelements of a switch arrangement according to an embodiment of the invention;

15 Figure 3b shows a simplified view of the switch in Figure 3a;

Figure 3c shows an unpopulated symmetric $LN \times LN$ matrix for the switch shown in Figure 3a;

20 Figure 4 shows steps in a method according to an embodiment of the invention;

Figure 5 is a sketch showing schematically the aggregation of requests for the switch shown in Figure 3a in the first level matching method according to an embodiment of the invention;

25 Figure 6 shows schematically the steps of aggregating requests and the 1st level of matching in multi-level matching scheme according to an embodiment of the invention;

Figure 7 shows schematically the steps of performing multiple, parallel matchings of N 30 elements, each having L input ports, including de-aggregation, in the 2nd level of a multi-level matching scheme according to the invention; and

Figure 8 shows the pointer positions for the 2×4 asymmetric request matrix $[r_1(n,j)]$ in the first cycle or frame ($k=0$) for a multi-level matching scheme according to an embodiment of 35 the invention.

The best mode of the invention as currently contemplated by the inventors will now be described. This invention relates to the matching part of a frame-based scheduling algorithm. The matching algorithm is able to use multiple levels of aggregation for packet requests. The term packet is used here to refer to multi-cast and unicast packets of fixed length (i.e., fixed as in a cell has a fixed length) or variable length as is apparent to those skilled in the art. The switch arrangements described relate to a number of possible embodiments, including packet, cell, and circuit switching arrangements. A cell switch can additionally include means to switch packets of fixed and/or variable length in some embodiments of the invention.

The invention can be used to match service requests in any switch arrangement provided over a network. For example a matching for service rate requests between ports on any switch can be provided by the invention, as well as a matching on a larger scale between sub-networks within a communications network. For example, the matching process can be used when traffic needs to travel between interconnecting optical networks and rings. As has been mentioned above, the invention can also, in some embodiments, be used to match service requests in a circuit switch environment.

- 20 A specific embodiment of the invention will now be described with reference to Figures 3a,3b, and 3c of the accompanying drawings. Figure 3a shows schematically a switch comprising a number of elements #a1,...,#aN and #b1,...,#bM between which traffic can be switched over switch fabric 31. Each of the elements #a1..#aN has a number of subelements, and each of the elements #b1..#bM has a number of subelements. The number of subelements L_1 may not be equal to the number of subelements L_2 , and the number of elements N may not equal the number of elements M in some embodiments of the invention. In a preferred embodiment of the invention the product L_1N is equal to the product L_2M , and in the best mode of the invention $N = M$ and $L_1 = L_2$.
- 30 The subelements may comprise unidirectional ingress or egress to the switch fabric, or they each may comprise bi-directional ingress and egress facilities to and from the switch fabric.

In one embodiment of the invention, the elements N, M comprise sub-networks in a network connected by a hub switch fabric, and each subelement comprises a node or

terminal on each sub-network #a1...#aN or sub-network #b1...#bM. For example, consider an embodiment where a switch is arranged to switch traffic moving between different rings and/or networks and needs to be capable of switching traffic at different levels of aggregation, for example between optical networks (particularly passive optical

5 networks PONs). Such a switch needs to have high performance and support fast switching speeds in a reliable and fair manner, as discussed by Bianco et al in their paper on Access Control Protocols for Interconnected WDM Rings in the DAVID Metro Network, the contents of which are hereby incorporated by reference. .

10 It will be appreciated by those skilled in the art that whilst Figure 3 is described with reference to elements and subelements, there is a clear analogy to embodiments in which the elements comprise, for example, different terminals or ports of a switch, or different terminals of a network, or different terminals of a sub-network, or a sub-network having a number of different terminals, or a switch having a number of terminals or ports.

15 In the prior art, it is known to perform the matching between the input and output ports or terminals of a switch or network conventionally in one operation or hierarchical level. The preferred embodiment of the invention proposes a matching scheme for a switching arrangement comprising a number of input subelements (for example ports or terminals)

20 which are grouped into elements and the matching is performed in more than one hierarchical level in a global, end-to-end manner.

The invention has the benefit of increasing the amount of parallel processing that can be performed in the matching and reducing the computing steps required for the matching.

25 The elements can, in some embodiments of the invention, be arbitrary sub-sets of the subelements, for example, the subelements could comprise ports (terminals) without any particular physical significance or in alternative embodiments comprise ports on real sub-networks. For example, if the switch input and output elements comprise rings in an interconnecting switching arrangement of rings, then the input and output subelements

30 could comprise the individual nodes or terminals. Alternatively, if the switching arrangement is a large switch comprising a plurality of interconnected smaller switches, then the elements could comprise the ports on smaller switches.

This invention therefore provides a global matching algorithm for use in either single-stage

35 or multi-stage switching and buffering networks, without resorting to complete autonomy

of the smaller elements (i.e. sub-sets of ports or terminals, switches or sub-networks), nor aggregating requests at too high a level (e.g. ring-ring or PON-PON). The invention provides a matching method in which the subelements (e.g., the ports or terminals) are grouped into elements and the matching is performed in more than one hierarchical level, 5 and in a global, end-to-end manner. This has the benefit of increasing the amount of parallel processing that can be performed in the matching and reducing the computing steps required for the matching.

As described with reference to the prior art, matching algorithms conventionally make use 10 of symmetric traffic request matrices between cell- or packet-switch ports, or between rings or PONs in packet networks, or between nodes or terminals in packet networks. But this invention employs asymmetric traffic request matrices, applied to different parts of the overall network with different levels of aggregation, in order to reduce the matching complexity. For example, at a first level of aggregation the matching may be between 15 input elements and output subelements, and a second level of matching may be between input subelements and output subelements. In this way, an asymmetric request matrix can be generated for service requests between an input switch element and an output port of an output switch element. Alternatively, an asymmetric matrix could be generated between an upstream ring element and a downstream node subelement or alternatively, 20 an upstream PON element and a downstream terminal subelement.

By providing a two-level, global (i.e., end to end) matching process between the elements and subelements and subelements to subelements, sufficient information about individual port, node or terminal identities can be retained to prevent receiver contentions and 25 source blocking, while reducing the overall matching complexity.

In some embodiments of the invention, more than one level of aggregation can be implemented in the matching method, and as such elements become subelements with respect to elements in a higher layer of matching. For example, it is possible for multiple 30 layers of matching are performed in a hierarchy of matching levels in some embodiments of the invention. As an example, one method of matching according to an embodiment of the invention comprises the following steps:

firstly, aggregating service requests to the highest level of the matching hierarchy, and

secondly, resolving contention for said service requests at the highest level of the matching hierarchy, and

thirdly, resolving contention in turn down through the matching levels to the lowest level of matching.

5

In Figure 3a, a switch 31 is shown schematically to be arranged to switch traffic between a number of elements N (denoted #a1...#aN), each having L_1 subelements across a suitable switch fabric 31 (for example a hub) to a number of output subelements, here ML_2 in number. Each element a#1..a#N comprises a number of different subelements L_1 , and in 10 Figure 3a, an embodiment of the invention is shown where the ML_2 subelements are shown aggregated into M groups of L_2 subelements. The grouping (or aggregation) of the output subelements into M elements, does not occur in other embodiments of the invention. Switch 31 therefore comprises NL_1 inputs and ML_2 outputs, i.e., switch 31 effectively comprises an $NL_1 \times ML_2$ switch. As an example, consider an optical 15 embodiment of the invention where aggregation of both inputs and outputs may be present if the N, M elements are PONS or optical ring networks as then both the L_1 and L_2 subelements may comprise user terminals or nodes.

Figure 3b shows a simplified representation of the switching arrangement showed in 20 Figure 3a, which illustrates more clearly the subelements forming the inputs and outputs of the switch 31. In Figure 3b, switching arrangement 40 comprises L_1N input subelements i (for example ports) and L_2M output subelements j (for example ports) j. A conventional matching algorithm for frame-based scheduling such as that which Bianco et al describe employs multiple phases for matching, such as was described referring to the 25 prior art. Such a conventional technique produces a match for a symmetric request matrix, $L_1N \times L_2M$ (or if $N = M$ and $L_1 = L_2 = L$, $LN \times LN$) in size such as is shown in Figure 3c (where a grid is shown as the matrix is not yet unpopulated).

Asymmetric, Multi-Stage Matching Using Multiple Levels of Aggregation

30 Referring now to Figure 4 of the accompanying drawings, an overview of the steps in a matching method according to the invention is shown suitable for the switch environment shown in Figures 3a, 3b, and 3c of the accompanying drawings.

Where a conventional input queued switch arrangement is being considered, the term 35 subelement is used to refer to any ports and the term element then refers to an

aggregation of such ports. Where the switch arrangement is provided by a network element interconnecting a number of optical networks (for example, an optical ring network, or passive optical networks (PONs), the term subelement is used to refer to any nodes or terminals on the PONs and the term element refers to an aggregation of such 5 nodes or terminals, for example, the term element could refer as such as to a ring network or a PON.

As has been mentioned, in some embodiments of the invention, the switch arrangement comprises part of a network, and the network comprises interconnected sub-networks. In 10 such embodiments, at one hierarchical level the sub-networks are the elements, and the nodes or terminals in each sub-network comprise the subelements. However, at a different hierarchical level, each node or terminal can comprise an element, and ports on the nodes or terminals can comprise the subelements. For example, where the switch arrangement is provided by a network element interconnecting a number of optical 15 networks (for example, passive optical networks (PONs), the term subelement is used to refer to any nodes or terminals on the PONs and the term element can refer to the PON. As will be appreciated by those skilled in the art, the hierarchical matching process according to the invention is not limited to such embodiments, but may be implemented in any switching environment where differing levels of aggregation can be effected at least 20 for the inputs to the switch arrangement.

In Figure 4, an algorithm according to one embodiment of the invention is shown in which NL_1 input subelements are capable of generating service requests for ML_2 output subelements over a switch fabric. The NL_1 input subelements are aggregated as N 25 elements $\#a1\ldots\#aN$, each element comprising L_1 subelements. The ML_2 output subelements may be aggregated into M elements, each element comprising L_2 subelements in some embodiments of the invention, but need not be so aggregated in other embodiments of the invention. In Figure 4, the L_2 subelements are aggregated into $M=N$ elements, $\#b1\ldots\#bM$. Each of the N elements $\#a1\ldots\#aN$ of the switch arrangement 30 has are initially aggregated together in step 41 by summing the total number of requests for each of the L_1 subelements of each of the N elements, i.e., for each element the total number of requests is summed over its L_1 input ports in step 41.

A first matching is then performed in which the service requests are matched at a first 35 aggregation level by generating an asymmetric $N \times ML_2$ request matrix for each of the N

input elements in step 42. The notation used here means that the matrix has N rows and ML_2 columns, where N is an integer and ML_2 is an integer.

A second matching process is then performed in step 43 in which N separate matchings

5 are performed, one for each of the N elements comprising L_1 subelements (input ports). This involves N separate $L_1 \times LM$ asymmetric request matrices. De-aggregation is thereby performed back from the aggregate level of the N elements to the ML_2 output subelements (i.e., output ports) to the aggregate level of L input ports to ML_2 output ports in step 44. It will be appreciated by those skilled in the art that the N matchings of step

10 43 could of course be performed sequentially, but it is advantageous in terms of the total time taken to run the algorithm if the number of computing steps (times) can be reduced by performing them simultaneously, in parallel, using multiple matching "processors". The latter is the preferred approach and is adopted in the best mode of the invention currently contemplated by the inventors. It is also possible for the number of elements and

15 subelements to differ on each side of the switch as has been mentioned before.

Step 41 Aggregation of Requests

Figure 5 shows schematically how each of the L_1 subelements of input element #a1 in Figure 3a is initially aggregated. Strictly speaking, because the particular matching

20 algorithm being used as an example takes the queue lengths as inputs for its first normalisation stage, the requests in this step are simply the queue lengths. (However, the number of requests could be any alternative number of requests for cells/packets to be switched, using other criteria for calculating that number. For example, each VOQ request used could be calculated as the queue length limited to a maximum of F requests for the

25 next frame).

Figure 6 shows the aggregation of request step 41 and the first level of matching step 42 of Figure 4 in more detail. In Figure 6, $N = M$, and $L_1 = L_2 = L$ for simplicity.

30 In Figure 6, the N sub-groups of L input ports each employs an $L \times LN$ asymmetric queue matrix $[Q(i,j)]_{\text{individual}}$ to represent the numbers of backlogged cells/packets in each input port destined for each output port. Each $L \times LN$ matrix $[Q(i,j)]_{\text{individual}}$ is simply that portion of the global $LN \times LN$ $[Q(i,j)]$ matrix for all VOQs of the entire switch or switching network relating to that particular sub-group. Thus each sub-group of input ports has all its input

35 port-output port requests recorded. The queue lengths of all input ports in each

$[Q(i,j)]_{\text{individual}}$ matrix are then aggregated (summed) into N off $1 \times LN$ aggregated queue matrices, which in effect are equivalent to one $N \times LN$ aggregated queue matrix $[Q(n,j)]_{\text{agg}}$ between the sub-groups and the output ports.

5 **1st Level of Matching**

The first level of matching is just one matching covering the entire switch or network. It takes the $N \times LN$ asymmetric, aggregated queue matrix $[Q(n,j)]_{\text{agg}}$ as its input, as shown in Figure 1.

10 Output and input booking using the example matching algorithm are summarised as follows. Outputs for the matching still represent the overall output ports of the switch, but inputs represent here the N sub-groups. The matrices now possess an index representing the 1st or 2nd level of matching.

15 Firstly, the aggregated queue matrix undergoes a normalisation stage:

$$[Q(n,j)]_{\text{agg}} \Rightarrow [Q_{\text{norm}}(n,j)]_{\text{agg}}, [r_1(n,j)]$$

Then the output booking and input booking phases described herein above are performed in a similar manner:

20 Output Booking phase: $[r_1(n,j)] \Rightarrow [g_1(n,j)]; \sum_n g_1(n,j) \leq F - [\sum_n Q_{\text{norm}}(n,j)]_{\text{agg}}$

Input Booking phase: $[g_1(n,j)] \Rightarrow [a_1(n,j)]; \sum_j a_1(n,j) \leq F$

Here the $g_1(i,j)$ are elements in the first matrix of granted requests and the $a_1(i,j)$ are elements in the first matrix of accepted grants.

25 **2nd Level of Matching**

Figure 7 shows in more detail steps 43 and 44 in which multiple, parallel matchings of N elements of input ports, including de-aggregation occurs in the second level of matching.

In the 2nd level of multi-level matching the aggregated acceptances in the acceptance matrix $[a_1(n,j)]$ from the 1st stage of matching provide the limits for matching between the

30 overall input and output ports within each of the N sub-groups of input ports. The input for the matching within each sub-group is taken as the original $L \times LN$ asymmetric queue matrix $[Q(i,j)]_{\text{individual}}$. Performing the matchings automatically provides de-aggregation back from sub-groups-output ports to input ports-output ports. Normalisation and output and

input booking is then performed in the manner described below. In this 2nd level, outputs and inputs of the matching once again represent the overall output and input ports of the switch/network.

5 Normalisation stage: $[Q(i,j)]_{\text{individual}} \Rightarrow [Q_{\text{norm}}(i,j)]_{\text{individual}}, [r_2(i,j)]$

Output Booking phase: $[r_2(i,j)] \Rightarrow [g_2(i,j)]; \sum_i g_2(i,j) \leq a_1(n,j) - [\sum_i Q_{\text{norm}}(i,j)]_{\text{individual}}$

Here the $g_2(i,j)$ are elements in the second matrix of granted requests and the $a_2(i,j)$ are elements in the second matrix of accepted grants.

10

The summation is taken over only the input ports within a sub-group, and the value of n in the summation is obviously the identity of that sub-group.

Input Booking phase: $[g_2(i,j)] \Rightarrow [a_2(i,j)]; \sum_j a_2(i,j) \leq F$

15 An example demonstrating a specific embodiment of the invention

Multi-level Matching –Aggregation

As a simple example of the hierarchical, multi-level matching principle, it is possible to partition the overall queue matrix $[Q(i,j)]$ for the switch in Eqn.1 into N=2 sub-groups of

20 L=2 input ports each and to aggregate the requests (queue lengths) destined from the same sub-group to each output port into a single 2x4 matrix, i.e.

25

$$\begin{bmatrix} 1 & 2 & 4 & 8 \\ 2 & 4 & 8 & 1 \end{bmatrix}$$

Eqn. 23 $[Q(i,j)]_{\text{individual}} \rightarrow [Q(n,j)]_{\text{agg}} = [\sum_i Q(i,j)]_{\text{individual}} = \begin{bmatrix} 3 & 6 & 12 & 9 \\ 12 & 9 & 3 & 6 \end{bmatrix}$

$$\begin{bmatrix} 4 & 8 & 1 & 2 \\ 8 & 1 & 2 & 4 \end{bmatrix}$$

Multi-Level Matching – the First -Level Matching Normalisation Stage

Because the input “ports” for this matrix are the sub-groups, the row-sums over sub-groups are obviously larger, typically, than the column sums over output ports. We will define maxval as the maximum column-sum or (row-sum/L), the latter being (row-sum/2)

5 in this example, because each sub-group contains L=2 input ports. With this definition, maxval is again 15, and every queue length is multiplied by the ratio F/15=4/15 and the integer part of the resulting number is taken. Hence the normalised queue matrix becomes

Eqn. 24

$$[Q_{\text{norm}}(n,j)]_{\text{agg}} = \begin{bmatrix} 0 & 1 & 3 & 2 \\ 3 & 2 & 0 & 1 \end{bmatrix}$$

10 All of these cells or packets are assumed already to be granted by the output ports and accepted by the sub-groups. The request matrix presented to the next “no overbooking” stage is the difference between the original queue matrix and the normalised queue matrix, i.e. the remaining requests

15 Eqn. 25 $[r_1(n,j)] = [Q(n,j)]_{\text{agg}} - [Q_{\text{norm}}(n,j)]_{\text{agg}}$

$$= \begin{bmatrix} 3 & 6 & 12 & 9 \\ 12 & 9 & 3 & 6 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 3 & 2 \\ 3 & 2 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 3 & 5 & 9 & 7 \\ 9 & 7 & 3 & 5 \end{bmatrix}$$

Multi-Level Matching – the First -Level Matching “No Overbooking” Stage - Output

20 Booking Phase

The number of requests in effect already granted by the output ports in the normalisation stage is

Eqn. 26 $\sum_n [Q_{\text{norm}}(n,j)]_{\text{agg}} = [3 \ 3 \ 3 \ 3]$

25 These are precisely the same numbers as were granted by the normalisation stage in conventional single-level matching (Eqn.4).

Once again, the remaining number of grants available in each output port is therefore

Eqn. 27 $[F \ F \ F \ F] - [3 \ 3 \ 3 \ 3] = [1 \ 1 \ 1 \ 1]$

30

Step 3 of the “no overbooking” algorithm described by Bianco et al applies again. But the NOB25 pointer up-date rule needs to be modified for the asymmetric request matrix $[r_1(n,j)]$, to ensure that input ports and output ports point to each other, even though there

are different numbers of each. (NB the N input ports are synonymous at this 1st matching level with the N sub-groups of overall input ports). Input and output ports for this 1st-level matching now require slightly different relationships between ports, i.e.

5 Multi-Level Matching – the First -Level Matching Pointer Up-Date Rule for Asymmetric Request Matrix:

$$\text{Eqn. 28 for input ports: } p_{\text{out}} = 1 + [(LN - P_{\text{in}} + K)_{\text{mod }} LN]$$

In our example with L=2 and N=2, this becomes

10

$$\text{Eqn. 29 } p_{\text{out}} = 1 + [(4 - P_{\text{in}} + k)_{\text{mod }} 4]$$

$$p_{\text{in}} = 1 + [(4 - P_{\text{out}} + k)_{\text{mod }} 2]$$

Figure 8 summarises the pointer positions for the 2x4 asymmetric request matrix $[r_1(n,j)]$ in
 15 the first cycle or frame ($k=0$). In any cycle (frame) k , two input ports point to two output ports, each of which output ports points back to the same input port that points to it. The remaining two output ports also point to the two input ports. Hence each input port is pointed to by two output ports, but only two of the four output ports are pointed to by input ports. After 4 cycles or frames, each input port has pointed to each output port once in
 20 turn, and each output port has pointed to each input port twice.

Hence in the first frame $k=0$, with $L=2$ and $N=2$ in this example, output port 1 points to input port 2, 2 points to 1, 3 points to 2 and 4 points to 1, i.e. the pointers point to requests $r(2,1)$, $r(1,2)$, $r(2,3)$ and $r(1,4)$ in Eqn.13. All of these matrix elements have more than one
 25 request, and because Eqn.27 allows only 1 more available grant for each output port, each of these four matrix elements will be granted one more request, i.e.

$$\text{Eqn. 30 additional output booking grants, } [g_1(n,j)] = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Multi-Level Matching – the First -Level Matching “No Overbooking” Stage - Input
 30 Booking Phase

From Eqn.24, the number of requests in effect already accepted by the input ports (really the 2 sub-groups of input ports) in the normalisation phase is

$$\text{Eqn. 31 } [\sum_j Q_{\text{norm}}(n,j)]_{\text{agg}} = \begin{bmatrix} 6 \\ 6 \end{bmatrix}$$

Because each sub-group contains 2 overall input ports, the remaining number of acceptances available in each sub-group is therefore

$$\text{Eqn. 32} \quad \begin{bmatrix} 2F \\ 2F \end{bmatrix} - \left[\sum Q_{\text{norm}}(n,j) \right]_{\text{agg}} = \begin{bmatrix} 8 \\ 8 \end{bmatrix} - \begin{bmatrix} 6 \\ 6 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

The matrix to be used in this input booking phase is the additional output booking grants matrix $[g_1(n,j)]$ (Eqn.30). Step 2 of the "no overbooking" algorithm applies. All of the additional grants are therefore accepted, so the additional acceptance matrix is

$$\text{Eqn. 33} \quad \text{additional input booking acceptances, } [a_{1\text{additional}}(n,j)] = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

The final 1st-level acceptance matrix becomes

$$\text{Eqn. 34} \quad [a_1(n,j)] = [Q_{\text{norm}}(n,j)]_{\text{agg}} + [a_{1\text{additional}}(n,j)] = \begin{bmatrix} 0 & 1 & 3 & 2 \\ 3 & 2 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 2 & 3 & 3 \\ 4 & 2 & 1 & 1 \end{bmatrix}$$

10 Note that all output ports fill all $F=4$ time slots and all input ports (sub-groups) fill all $2F=8$ time slots, in this first frame. A full set of 16 cells or packets are accepted.

Multi-Level Matching – the Second Level Matching for Sub-Group 1

From Eqn.23 the queue matrix for the first sub-group is

$$15 \quad \text{Eqn. 35} \quad [Q(i,j)]_{\text{individual}} = \begin{bmatrix} 1 & 2 & 4 & 8 \\ 2 & 4 & 8 & 1 \end{bmatrix}$$

and from Eqn34 the number of acceptances to each output port from the 1st level of matching are

$$\text{Eqn. 36} \quad [a_{1,1}(j)] = [0 \ 2 \ 3 \ 3]$$

These acceptances represent the maximum number of grants that will be allowed by 20 each output port to all input ports in this 2nd level of matching.

Multi-Level Matching – the Second Level Matching for Sub-Group 1 Normalisation Stage

The maximum row-sum or column-sum, maxval, in Eqn.35 is 15. The normalised queue matrix $[Q_{\text{norm}}(i,j)]_{\text{individual}}$ is obtained from Eqn.35 by multiplying each element by the factor

25 $F/\text{maxval} = 4/15$ and taking the integer part of the resulting number, i.e..

$$\text{Eqn. 37} \quad [Q_{\text{norm}}(i,j)]_{\text{individual}} = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

All of these cells or packets are assumed already to be granted by the output ports and accepted by the input ports. The request matrix presented to the next "no overbooking" stage is the difference between the original queue matrix and the normalised queue

30 matrix, i.e. the remaining requests

$$\text{Eqn. 38 } [r_2(i,j)] = [Q(i,j)]_{\text{individual}} - [Q_{\text{norm}}(i,j)]_{\text{individual}} = \begin{bmatrix} 1 & 2 & 4 & 8 \\ 2 & 4 & 8 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 & 6 \\ 2 & 3 & 6 & 1 \end{bmatrix}$$

Multi-Level Matching – the Second Level Matching for Sub-Group 1 “No Overbooking”

5 Stage - Output Booking Phase

The number of requests in effect already granted by the output ports in the normalisation stage is

$$\text{Eqn. 39 } \left[\sum_i Q_{\text{norm}}(i,j) \right]_{\text{individual}} = [0 \ 1 \ 3 \ 2]$$

The number of additional grants allowed is

10

$$\text{Eqn. 40}$$

$$[a_1(1,j)] - \left[\sum_i Q_{\text{norm}}(i,j) \right]_{\text{individual}} = [0 \ 2 \ 3 \ 3] - [0 \ 1 \ 3 \ 2] = [0 \ 1 \ 0 \ 1]$$

Step 3 of the “no overbooking” algorithm applies. The pointer positions in the first cycle or frame ($k=0$) are as in Figure 8. Output port 2 points to input port 1 and output port 4 also points to input port 1. Both additional grants are made, i.e.

$$\text{Eqn. 41} \quad \text{additional output booking grants, } [g_2(i,j)] = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Multi-Level Matching – the Second Level Matching for Sub-Group 1 “No Overbooking”

Stage - Input Booking Phase

20 The number of requests in effect already accepted by the input ports in the normalisation stage is

$$\text{Eqn. 42} \quad \left[\sum_j Q_{\text{norm}}(i,j) \right]_{\text{individual}} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

The number of additional acceptances allowed is

$$\text{Eqn. 43} \quad \begin{bmatrix} F \\ F \end{bmatrix} - \left[\sum_j Q_{\text{norm}}(i,j) \right]_{\text{individual}} = \begin{bmatrix} 4 \\ 4 \end{bmatrix} - \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

25 The request matrix for this input booking phase is the additional output booking grants matrix $[g_2(i,j)]$ (Eqn.41). Step 3 of the “no overbooking” algorithm applies. In the first cycle or frame input port 1 points to output port 4 (Figure 8), so the additional input booking acceptance matrix becomes

$$\text{Eqn. 44} \quad [a_{2\text{additional}}(i,j)] = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The final acceptance matrix is the sum of the acceptances from the initial normalisation (Eqn.37) plus these additional acceptances from the "no overbooking" algorithm (Eqn.44), i.e.

Eqn. 45

$$\begin{aligned}
 [a_2(i,j)]_{i=1,2} &= [Q_{\text{norm}}(i,j)]_{\text{individual}} + [a_{2\text{additional}}(i,j)] \\
 5 &= \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\
 &= \begin{bmatrix} 0 & 0 & 1 & 3 \\ 0 & 1 & 2 & 0 \end{bmatrix}
 \end{aligned}$$

10 Multi-Level Matching – the Second Level Matching for Sub-Group 2

From Eqn.23 the queue matrix for the second sub-group is

Eqn. 46

$$[Q(i,j)]_{\text{individual}} = \begin{bmatrix} 4 & 8 & 1 & 2 \\ 8 & 1 & 2 & 4 \end{bmatrix}$$

and from Eqn.34 the number of acceptances to each output port from the 1st level of matching are

15 Eqn. 47

$$[a_1(2,j)] = [4 \ 2 \ 1 \ 1]$$

These acceptances represent the maximum number of grants that will be allowed by each output port to all input ports in this 2nd level of matching.

20 Multi-Level Matching – the Second Level Matching for Subgroup 2 Normalisation Stage

The maximum row-sum or column-sum, maxval, in Eqn.46 is 15. The normalised queue matrix $[Q_{\text{norm}}(i,j)]_{\text{individual}}$ is obtained from Eqn.46 by multiplying each element by the factor $F/\text{maxval} = 4/15$ and taking the integer part of the resulting number, i.e.

Eqn. 48

$$[Q_{\text{norm}}(i,j)]_{\text{individual}} = \begin{bmatrix} 1 & 2 & 0 & 0 \\ 2 & 0 & 0 & 1 \end{bmatrix}$$

25 All of these cells or packets are assumed already to be granted by the output ports and accepted by the input ports. The request matrix presented to the next "no overbooking" stage is the difference between the original queue matrix and the normalised queue matrix, i.e. the remaining requests

30 Eqn. 49

$$\begin{aligned}
 [r_2(i,j)] &= [Q(i,j)]_{\text{individual}} - [Q_{\text{norm}}(i,j)]_{\text{individual}} \\
 &= \begin{bmatrix} 4 & 8 & 1 & 2 \\ 8 & 1 & 2 & 4 \end{bmatrix} - \begin{bmatrix} 1 & 2 & 0 & 0 \\ 2 & 0 & 0 & 1 \end{bmatrix} \\
 &= \begin{bmatrix} 3 & 6 & 1 & 2 \\ 6 & 1 & 2 & 3 \end{bmatrix}
 \end{aligned}$$

Multi-Level Matching – the Second Level Matching for Sub-Group 2 “No Overbooking”
Stage - Output Booking Phase

The number of requests in effect already granted by the output ports in the normalisation
 5 stage is

Eqn. 50 $[\sum_i Q_{\text{norm}}(i,j)]_{\text{individual}} = [3 \ 2 \ 0 \ 1]$

The number of additional grants allowed is

Eqn. 51 $[a_1(2,j)] - [\sum_i Q_{\text{norm}}(i,j)]_{\text{individual}} = [4 \ 2 \ 1 \ 1] - [3 \ 2 \ 0 \ 1] = [1 \ 0 \ 1 \ 0]$

10 Step 3 of the “no overbooking” algorithm applies. The pointer positions in the first cycle or frame are as in Figure 8. Output port 1 points to input port 2 and output port 3 also points to input port 2. Both additional grants are made, i.e.

Eqn. 52 additional output booking grants, $[g_2(i,j)] = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}$

15 Multi-Level Matching – the Second Level Matching for Sub-Group 2 “No Overbooking”
Stage - Input Booking Phase

The number of requests in effect already accepted by the input ports in the normalisation stage is

Eqn. 53 $[\sum_j Q_{\text{norm}}(i,j)]_{\text{individual}} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$

20 The number of additional acceptances allowed is

Eqn. 54 $\begin{bmatrix} F \\ F \end{bmatrix} - [\sum_j Q_{\text{norm}}(i,j)]_{\text{individual}} = \begin{bmatrix} 4 \\ 4 \end{bmatrix} - \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

The request matrix for this input booking phase is the additional output booking grants matrix $[g_2(i,j)]$ (Eqn.52). Step 3 of the “no overbooking” algorithm applies. In the first cycle or frame input port 2 points to output port 3 (Figure 8), so the additional input booking

25 acceptance matrix becomes

Eqn. 55 $[a_{2\text{additional}}(i,j)] = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

The final acceptance matrix is the sum of the acceptances from the initial normalisation (Eqn.48) plus these additional acceptances from the “no overbooking” algorithm (Eqn.55), i.e.

30 Eqn. 56 $[a_2(i,j)]_{i=3,4} = [Q_{\text{norm}}(i,j)]_{\text{individual}} + [a_{2\text{additional}}(i,j)]$
 $= \begin{bmatrix} 1 & 2 & 0 & 0 \\ 2 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

$$= \begin{bmatrix} 1 & 2 & 0 & 0 \\ 2 & 0 & 1 & 1 \end{bmatrix}$$

5 Multi-Level Matching – Overall Acceptance Matrix

The overall matrix of accepted requests is the concatenation of Eqn.45 and Eqn.56 for the two sub-groups, i.e.

Eqn. 57

$$[a_2(i,j)] = \begin{bmatrix} 0 & 0 & 1 & 3 \\ 0 & 1 & 2 & 0 \\ 1 & 2 & 0 & 0 \\ 2 & 0 & 1 & 1 \end{bmatrix}$$

Note that the 2nd level of matching has been unable to maintain the full set of 16 acceptances achieved by the 1st level of matching of the sub-groups. Only 14 cell or packet requests are finally accepted. This contrasts with the acceptance matrix resulting from conventional, single-level matching in Eqn.22, which achieves a full set of 16 acceptances. The reason for the reduction in accepted requests may be due to the nature of the modified NOB25 pointer up-date rule suggested for use with asymmetric request matrices in Eqn.28. This is discussed below, and a further modification is proposed.

Multi-Level Matching – Pointer Up-Date Rules for Asymmetric Request Matrices

For the particular asymmetric request matrices used in the working example, there appears to be a problem with the modified NOB25-like pointer up-date rule (Eqn.28). It can be shown that there are pairs of output ports that always point to just one, common, input port in every frame. (The particular input port for a given pair of output ports changes cyclically from frame to frame). This can result in requests being granted by two output ports to the same input port during the output booking phase, causing overbooking, which causes one of these grants to be dropped during the following input booking phase. This never happens for symmetric request matrices, because output port pointers never point to the same input port; there are enough input ports for all the output port pointers to point to different input ports. The problem has been found to happen when only two of the output ports are able to grant additional requests, and the two ports happen to be a pair that point to the same input port.

30

One embodiment of the invention provides a possible solution to this problem by deciding the pointer positions after the number m of output ports that are allowed to make additional grants is known. This would no longer be a deterministic pointer up-date rule.

However, the nature of the no overbooking algorithm described above ("NOB25") could be preserved to some extent by adapting the rule to the variable number of output ports m . Once this number m is known for a matching in a frame, these m ports would be ranked in order and the output port pointer positions would be calculated as follows

5

Eqn. 58 for output ports: $p_{in} = 1 + [(m - P_{out} + k)_{mod} L]$

where P_{out} is now the rank of the output port allowed to make additional grants, not the output port's identity. If the same sub-set of output ports are allowed to make additional 10 grants recurrently in each frame, then the overall effect may be like NOB25, in that the pointers would step around these ports by one port in each frame. In the worked example, if the same two output ports were to recur in every frame, they would always point to different input ports, which should remove the output overbooking.

15 Because there are always more output ports than input ports in the asymmetric request matrices, so that no two input ports ever need to point to the same output port, there is no need to adapt the NOB25 rule for the input ports. For these ports the pointer up-date rule can remain deterministic. The input port pointer positions would be calculated as in Eqn.28, i.e.

20

Eqn. 59 for input ports: $p_{out} = 1 + [(LN - P_{in} + k)_{mod} LN]$

where P_{in} is the input port's identity.

25 Using this rule, it can be shown that the overall acceptance matrix for the worked example using multi-level matching becomes

Eqn. 60 $[a_2(i,j)] = \begin{bmatrix} 0 & 0 & 1 & 3 \\ 0 & 2 & 2 & 0 \\ 1 & 2 & 1 & 0 \\ 3 & 0 & 0 & 1 \end{bmatrix}$

The matching now accepts a full set of 16 requests in the first frame. They are not all taken from the longest VOQs. For example, $a_2(3,3)$ is one of the shortest queues with only 30 one request.

Those skilled in the art will appreciate that the number of hierarchical matching levels depends on the number of levels of aggregation used. In the examples discussed with

reference to Figures 3 to 8 of the accompanying drawings, only two levels of aggregation were used. For larger switch arrangements it is possible to have more than two levels of aggregation and the number of matching levels increases accordingly.

- 5 The invention can be applied to switching arrangements having bi-directional elements/subelements as is apparent to those skilled in the art. The invention can be implemented in any suitable form, including as a suite of one or more computer programs which may be implemented in using software and/or hardware and the matching algorithm may be provided in a form which is distributed amongst several components.

10

The matching process can thus be implemented by one or more hardware and/or software components arranged to provide suitable means. For example, to implement the matching process on requests for service which are queued at the input of the input queued switch, the hardware and/or software component implementing the invention may include contain arbiters of parallel or serial operation.

15 Those skilled in the art will also realise that where reference has been made to the switch arrangement having NL inputs and NL outputs, this is a generalisation of the case where a switch arrangement has NL₁ inputs and NL₂ outputs, and that specific features of 20 such embodiments are not limited to the specified number of inputs and outputs which may have been simplified for simplicity.

CLAIMS

1. A matching method for a number N of first elements, each first element arranged to at least provide ingress to a switch arrangement, each of the first N elements comprising a number L_1 of first subelements, the switch arrangement having a number ML_2 of second subelements arranged to at least provide egress from said switch arrangement, and wherein each of the first L_1 subelements is capable of conveying a service request for at least one of said second subelements ML_2 , wherein the method comprises:
 - 10 firstly, for every one of the N first elements, aggregating service requests from all L_1 first subelements to each of the ML_2 second subelements, and
 - secondly, resolving contention for said service requests from all N first elements to one or more of said second ML_2 subelements, and
 - thirdly, for each first element, resolving contention between the L_1 subelements and said second ML_2 subelements.
- 20 2. A matching method as claimed in claim 1, wherein the step of resolving contention between the L_1 subelements and said second ML_2 subelements is performed in parallel for each said first element.
- 25 3. A matching method as claimed in claim 1 or 2, wherein the ML_2 second subelements of the switch arrangement are provided as a number M of second elements, each of said M second elements being associated with a number L_2 of second subelements.
4. A matching method as claimed in any previous claim, wherein each subelement is capable of generating at least one said service request.
- 30 5. A matching method as claimed in any one previous claim, wherein said first subelements and said second subelements are bi-directional and provide both ingress and egress from the switch fabric.
6. A matching method as claimed in claim 5, wherein said first subelements comprise said second subelements.

7. A matching method as claimed in any one of claims 1 to 4, wherein said first subelements and said second subelements are unidirectional and said first subelements provide ingress and said second subelements provide egress from the switch arrangement.

5

8. A matching method as claimed in any preceding claim, wherein said first and second subelements comprise ports in the switch arrangement and said first elements comprise aggregations of said first subelements.

10

9. A matching method as claimed in any one of claims 3 to 7, wherein said wherein said first and second subelements comprise ports in the switch arrangement, said first elements comprise aggregations of said first subelements and said second elements comprise aggregations of said second subelements.

15

10. A matching method as claimed in any one preceding claim, wherein said switch arrangement comprises an input queued cell switch and said service requests comprise requests for transmitting a service information rate from one of said first subelements to at least one of said second subelements.

20 11. A matching method as claimed in any one preceding claim, wherein said switch arrangement comprises an input queued cell switch and said service requests comprise requests for transmitting at least one cell from one of said first subelements to at least one of said second subelements.

25

12. A matching method as claimed in any one of claims 1 to 9, preceding claim, wherein said switch arrangement comprises an input queued packet switch and said service requests comprise requests for transmitting a service information rate from one of said first subelements to at least one of said second subelements.

30

13. A matching method as claimed in any one of claims 1 to 10, wherein said switch arrangement comprises an input queued packet switch and said service requests comprise requests for transmitting at least one packet from one of said first subelements to at least one of said second subelements.

35

14. A matching method as claimed in claim 12, wherein the packets have a fixed-length and comprise cells and wherein said packet switch is an input queued cell switch arranged to switch fixed-length cells, and said service requests comprise requests for transmitting one or more fixed-size cells from one of said first subelements to one or more of said second subelements.

15. A matching method as claimed in claim 13, wherein the packets have a fixed-length and comprise cells and wherein said packet switch is an input queued cell switch arranged to switch fixed-length cells, and said service requests comprise requests for transmitting a service information rate from one of said first subelements L_1 to one or more of said second subelements L_2 .

16. A method as claimed in any one of claims 1 to 9, wherein said switch arrangement comprises a circuit based switch and said service request comprises a request for a connection in a circuit-based switch.

17. A method as claimed in any one of claims 1 to 9, wherein said switch arrangement comprises a circuit based switch and said service request comprises a request for a bandwidth in a circuit-based switch.

18. A method as claimed in any one of claims 1 to 9, wherein said switch arrangement comprises a circuit based switch and said service request comprises a request for a service information rate in a circuit-based switch.

19. A method as claimed in any one of claims 10, 12, 15 or 16, wherein the service information rate is a bit rate.

20. A method as claimed in any one of claims 16 to 18, wherein said circuit based switch comprises at least one switch taken from the group consisting of:

any known time-domain, frequency domain, wavelength domain or space domain switching technologies.

21. A method as claimed in claim 20, wherein said circuit-based switch comprises a combination of said switches.

22. A method as claimed in any previous claim, wherein the switch arrangement comprises a network, and wherein said elements comprise aggregations of network terminals or nodes and said subelements comprise network terminals or nodes.

5

23. A method as claimed in any previous claim, wherein the switch arrangement comprises an arrangement of inter-connectable sub-networks, where said elements comprise sub-networks and said subelements comprise network terminals or nodes.

10 25. A method as claimed in claim 20, wherein said network is an optical network.

26. A method as claimed in claim 21, wherein said sub-networks comprise optical networks.

15 27. A method as claimed in any previous claim, wherein elements become subelements with respect to elements in a higher layer of matching.

28. A method as claimed in any previous claim, wherein multiple layers of matching are performed in a hierarchy of matching levels.

20

29. A method as claimed in any previous claim, wherein the method of matching comprises:

firstly, aggregating service requests to the highest level of the matching hierarchy, and

25 secondly, resolving contention for said service requests at the highest level of the matching hierarchy, and

thirdly, resolving contention in turn down through the matching levels to the lowest level of matching.

30 30. A matching method for a switch arrangement comprising a plurality N of input elements, each input element comprising a plurality (L_1) of input subelements, and a plurality M of output elements, each output element comprising a plurality L_2 of output subelements, the matching method comprising the following steps:

35 performing a first matching across the switch fabric for each of the plurality of N input elements and the L_2 subelements by performing the steps of:

summing a number of requests from each of the L_1 subelements of the input element;

generating a first $N \times ML_2$ request matrix;

matching the first request matrix to generate a first matrix of accepted requests;

5 and

performing a second matching across the switch fabric for each of the N input elements by performing the steps of:

generating N asymmetric second $L_1 \times ML_2$ matrices, for each of the N input elements; and

10 matching each of the N asymmetric second matrices to generate N second matrices of accepted requests; and

generating a $NL_1 \times ML_2$ matrix of accepted requests from the first $N \times ML_2$ matrix of accepted requests and the N second $L_1 \times ML_2$ accepted request matrices.

15 31. A matching method as claimed in claim 25, wherein the $NL_1 \times ML_2$ matrix of requests is symmetric.

32. A matching method as claimed in claim 26, wherein L_1 is equal to L_2 and N is equal to M .

20 33. A matching method as claimed in any previous claim, wherein said subelements comprise nodes in an optical ring network.

34. A matching method as claimed in any one previous claims, wherein said

25 subelements comprise terminals in a passive optical network.

35. A matching method as claimed in any one of claims 30 to 34, wherein the switch arrangement comprises a packet switch arrangement.

30 36. A matching method as claimed in claim 35, wherein the packet switch arrangement is capable of switching fixed-length packets.

37. A matching method as claimed in any one of claims 30 to 34, wherein the switch arrangement comprises a cell switching arrangement.

38. A matching method as claimed in claim 37, wherein the cell switching arrangement is capable of switching packets.

39. A switch arrangement, the switch arrangement having number N of first elements, each first element arranged to at least provide ingress to a switch arrangement, each of the first N elements comprising a number L_1 of first subelements, the switch arrangement having a number ML_2 of second subelements arranged to at least provide egress from said switch arrangement, and wherein each of the first L_1 subelements is capable of conveying a service request for at least one of said second subelements ML_2 , 10 wherein said service requests are conveyed by performing a matching method which comprises:

firstly, for every one of the N first elements, aggregating service requests from all L_1 first subelements to each of the ML_2 second subelements, and

15 secondly, resolving contention for said service requests from all N first elements to one or more of said second ML_2 subelements, and

thirdly, for each first element, resolving contention between the L_1 subelements and said second ML_2 subelements.

40. A switch arrangement as claimed in claim 39, wherein said matching method is 20 as claimed in any one of claims 1 to 38..

41. A network including a switch arrangement as claimed in any one of claims 39 or 40.

25 42. A suite of at least one computer programs arranged when executed to implement steps in a method according to any one of claims 1 to 38.

43. A suite of at least one computer programs as claimed in claim 42, wherein at least one program is arranged to be implemented by software running on a suitable 30 computational device.

44. A suite of at least one computer programs as claimed in claim 42 or 43, wherein at least one program is arranged to be implemented by suitably configured hardware.

45. A scheduler for a switching arrangement, the scheduler arranged to perform a scheduling process, the scheduling process comprising:

- a matching method as claimed in any one of claims 1 to 38; and
- a time-slot assignment process.

5

46. A matching scheme as claimed in any of claims 1 to 38, wherein the subelements comprise ports, and the matching updates the pointers to input ports according to the following rule: $p_{out} = 1 + [(LN - P_{in} + k)_{mod LN}]$ and the output ports are updated according to the following rule: $p_{in} = 1 + [(LN - P_{out} + k)_{mod L}]$

10

47. A matching scheme as claimed in any of claims 1 to 38, wherein the subelements comprise ports, and the matching updates the pointers to input ports according to the following rule: $p_{out} = 1 + [(LN - P_{in} + k)_{mod LN}]$ and the output ports are updated according to the following rule: $p_{in} = 1 + [(m - P_{out} + k)_{mod L}]$

15

ABSTRACT
MULTI-STAGE MATCHING PROCESS

A matching method for a number N of first elements, each first element arranged to at least provide ingress to a switch arrangement, each of the first N elements comprising a 5 number L_1 of first subelements, the switch arrangement having a number ML_2 of second subelements arranged to at least provide egress from said switch arrangement, and wherein each of the first L_1 subelements is capable of conveying a service request for at least one of said second subelements ML_2 , wherein the method comprises:

firstly, for every one of the N first elements, aggregating service requests from all 10 L_1 first subelements to each of the ML_2 second subelements, and

secondly, resolving contention for said service requests from all N first elements to one or more of said second ML_2 subelements, and

thirdly, for each first element, resolving contention between the L_1 subelements and said second ML_2 subelements.

15

Figure (4)

1/10

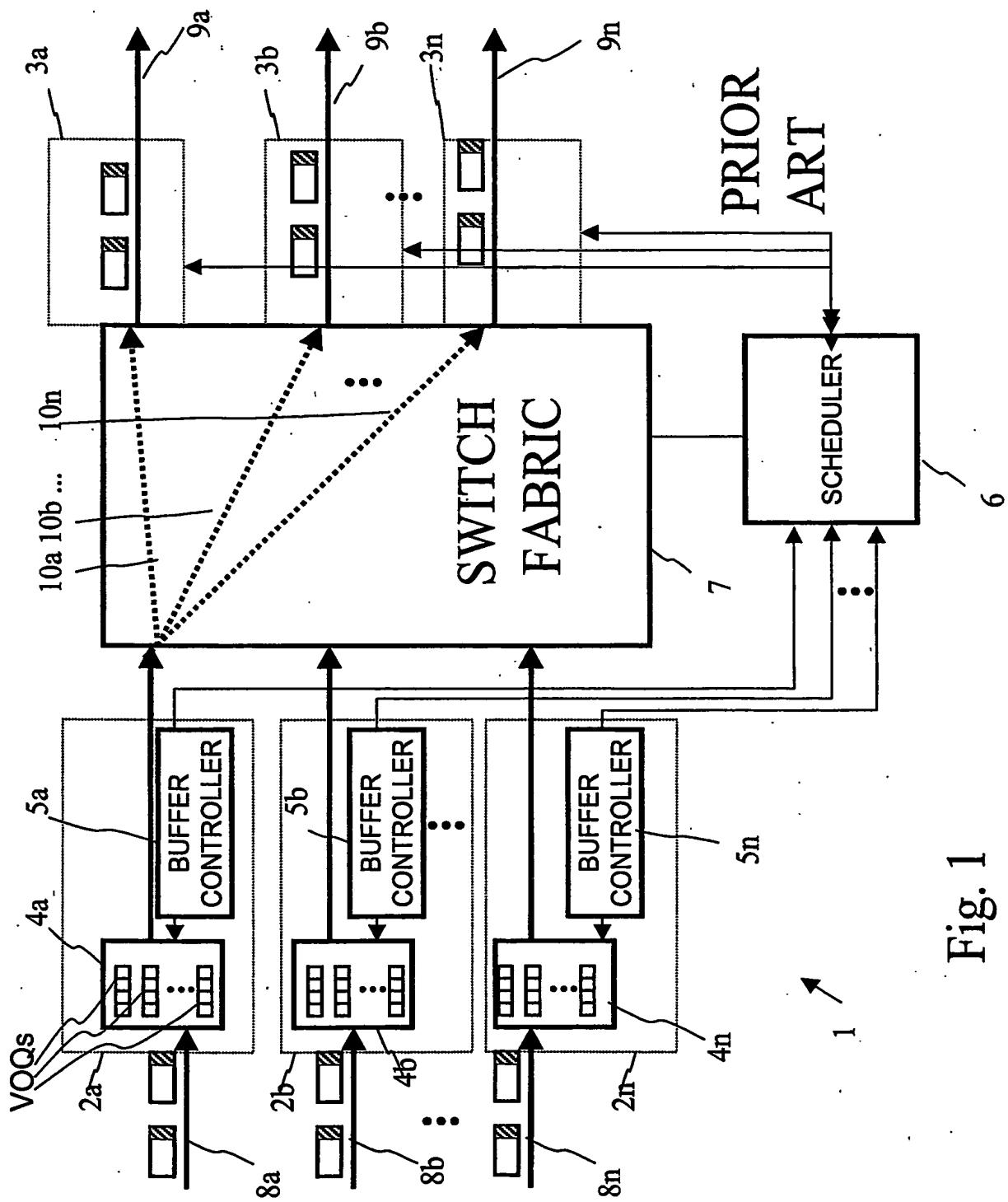
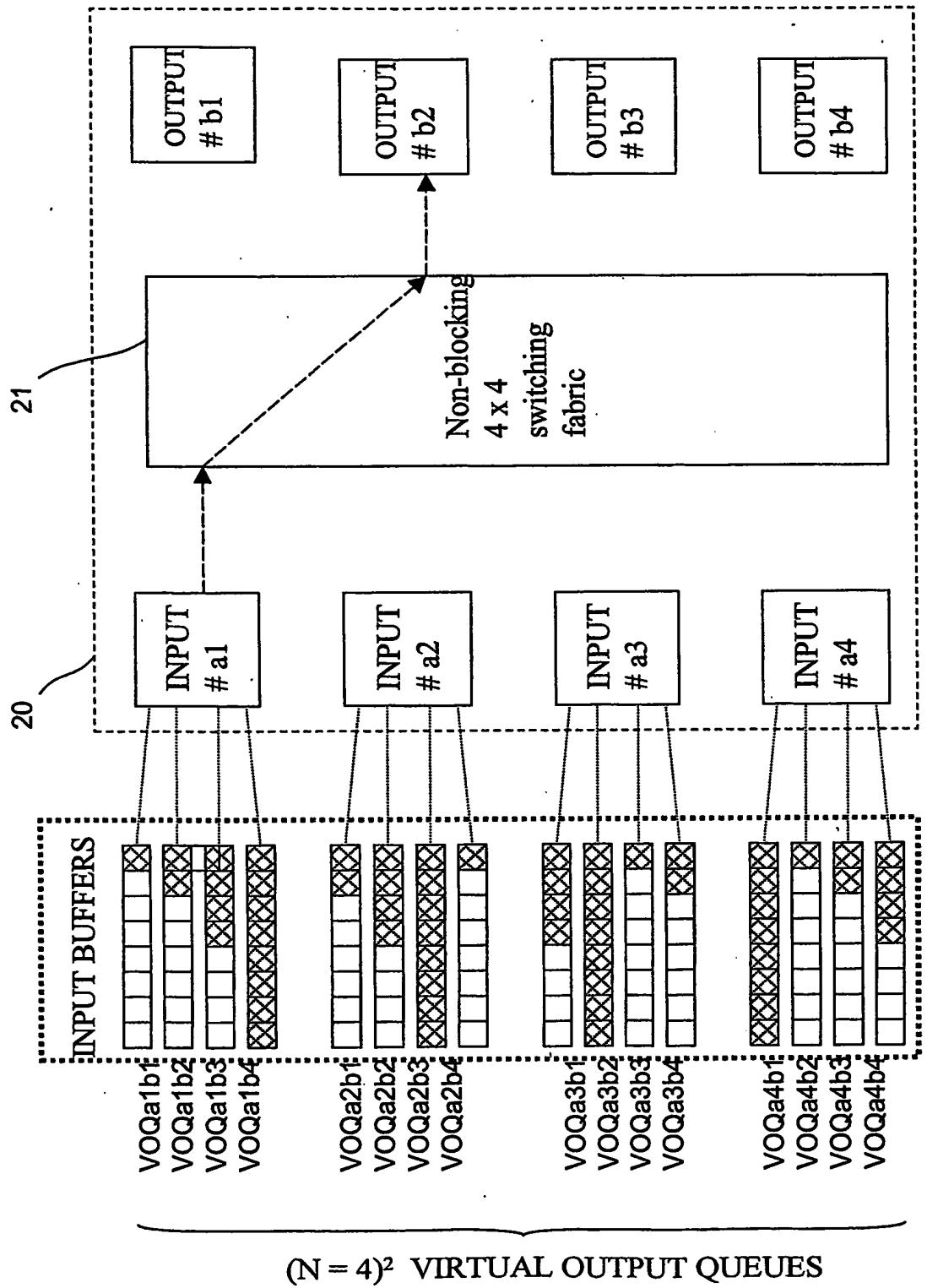


Fig. 1



PRIOR ART

Fig. 2

$$\begin{bmatrix} 1 & 2 & 4 & 8 \\ 2 & 4 & 8 & 1 \\ 4 & 8 & 1 & 2 \\ 8 & 1 & 2 & 4 \end{bmatrix}$$

3/10

L_2M outputs

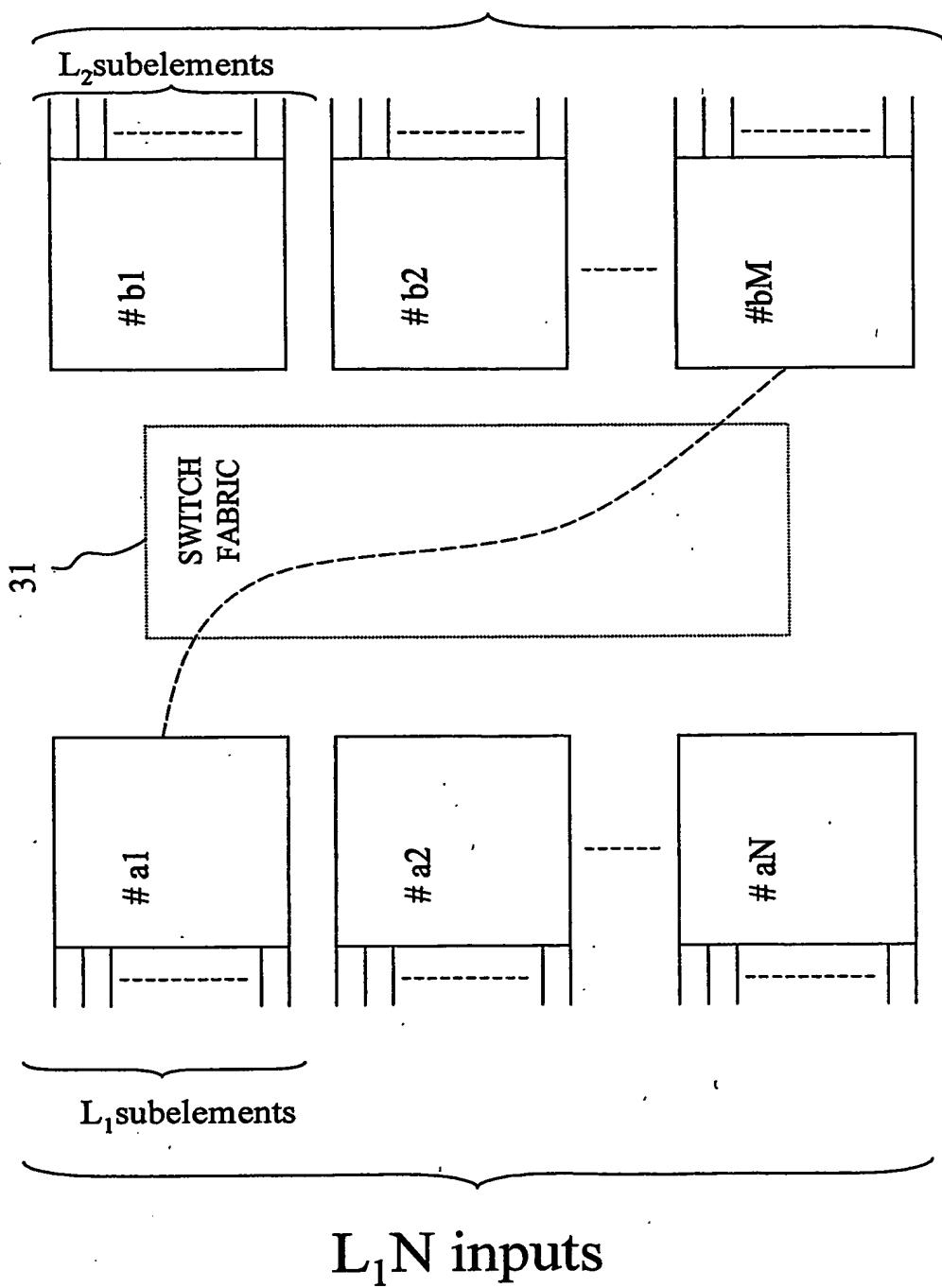


FIGURE 3a

4/10

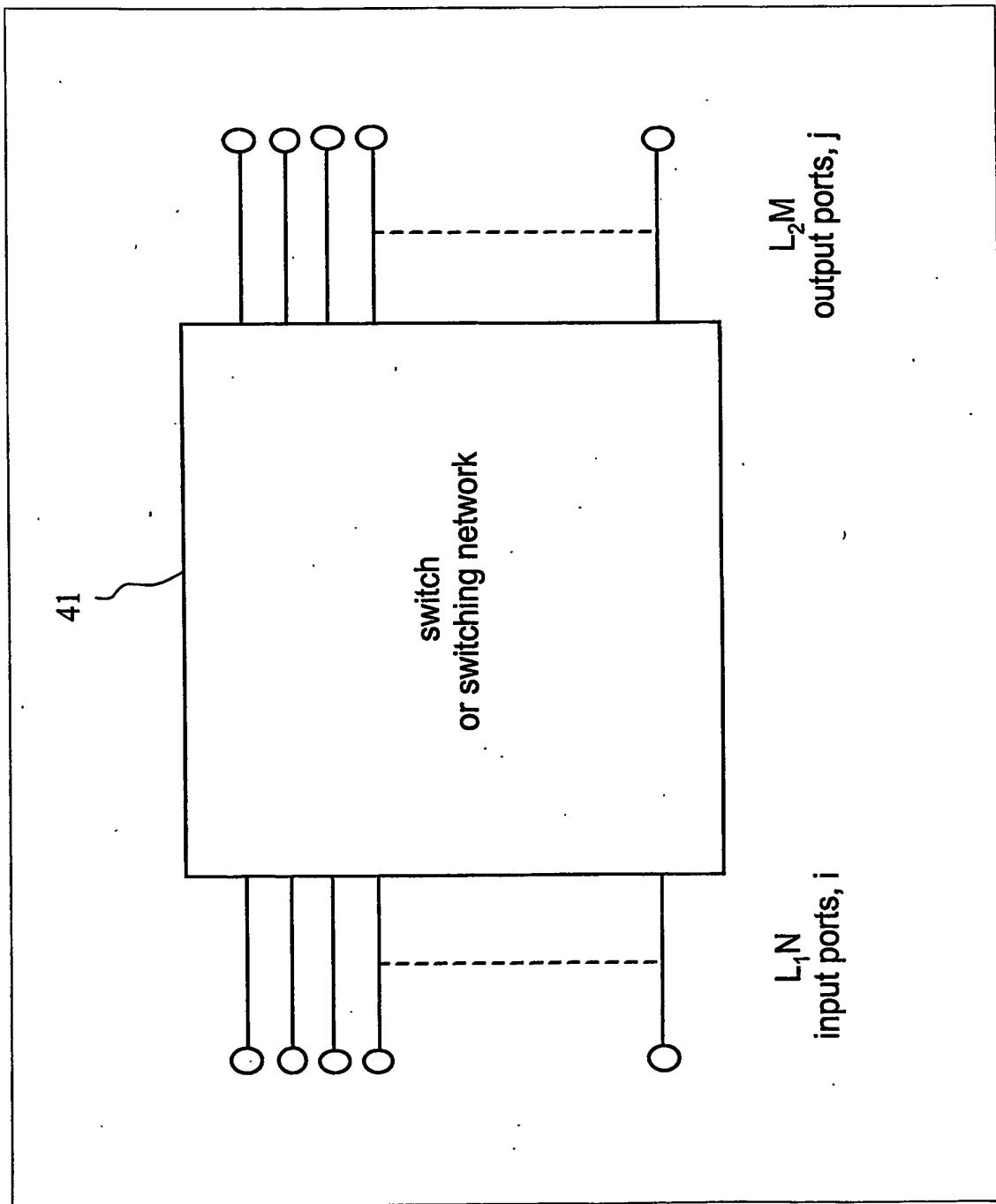


FIG. 3b

5/10

(UNPOPULATED)SYMMETRIC REQUEST MATRIX

$L_1 N$
inputs, i

$L_2 M$ outputs, j

FIG 3c

6/10

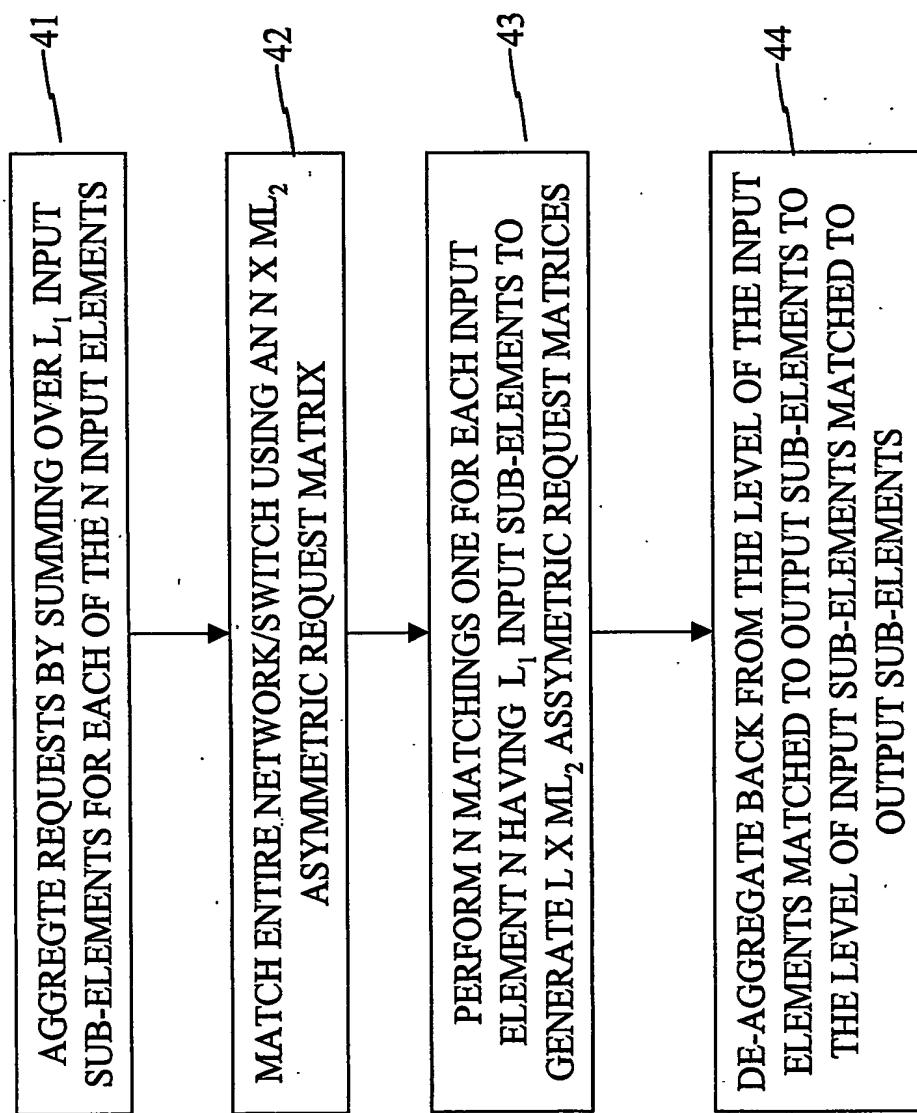
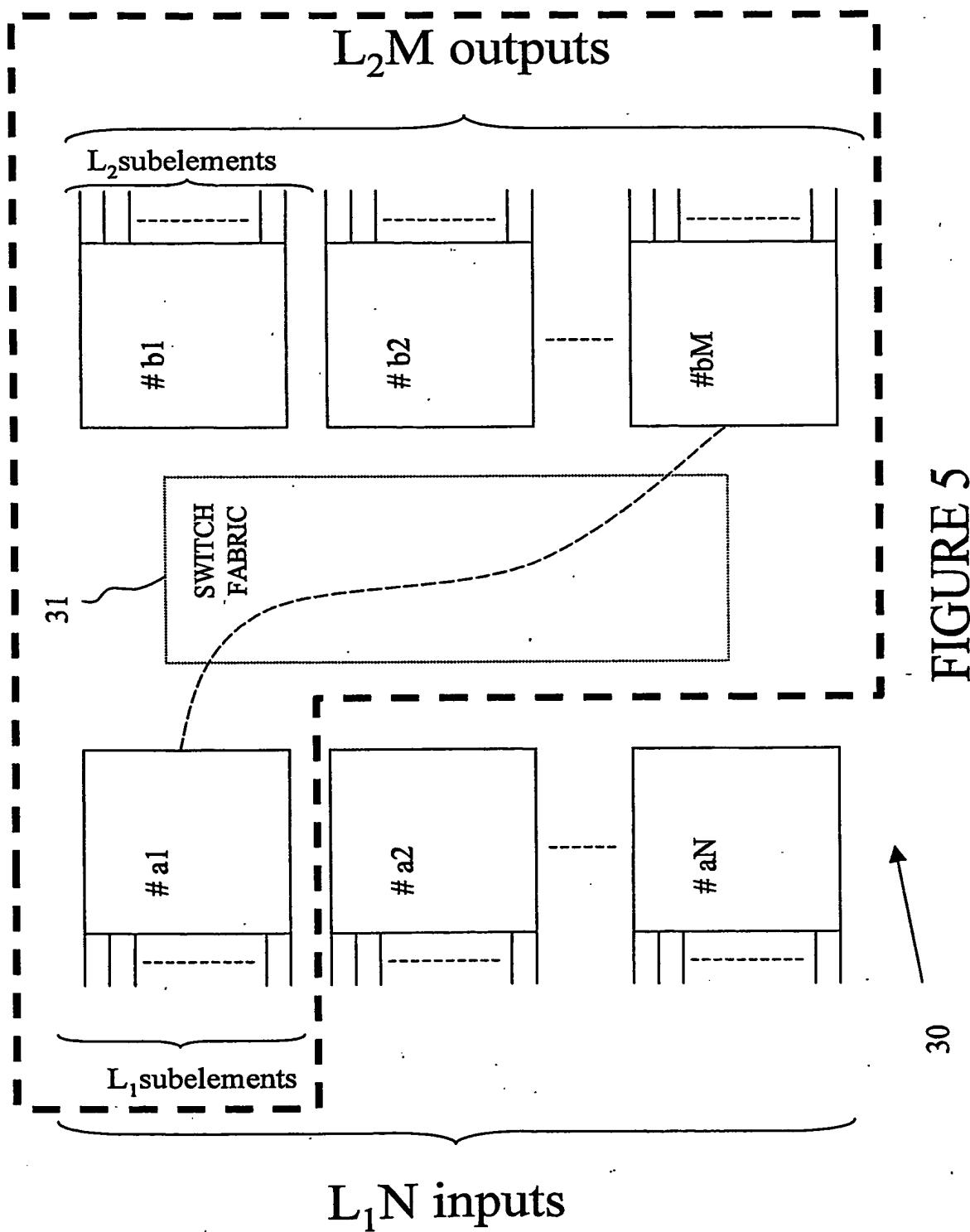
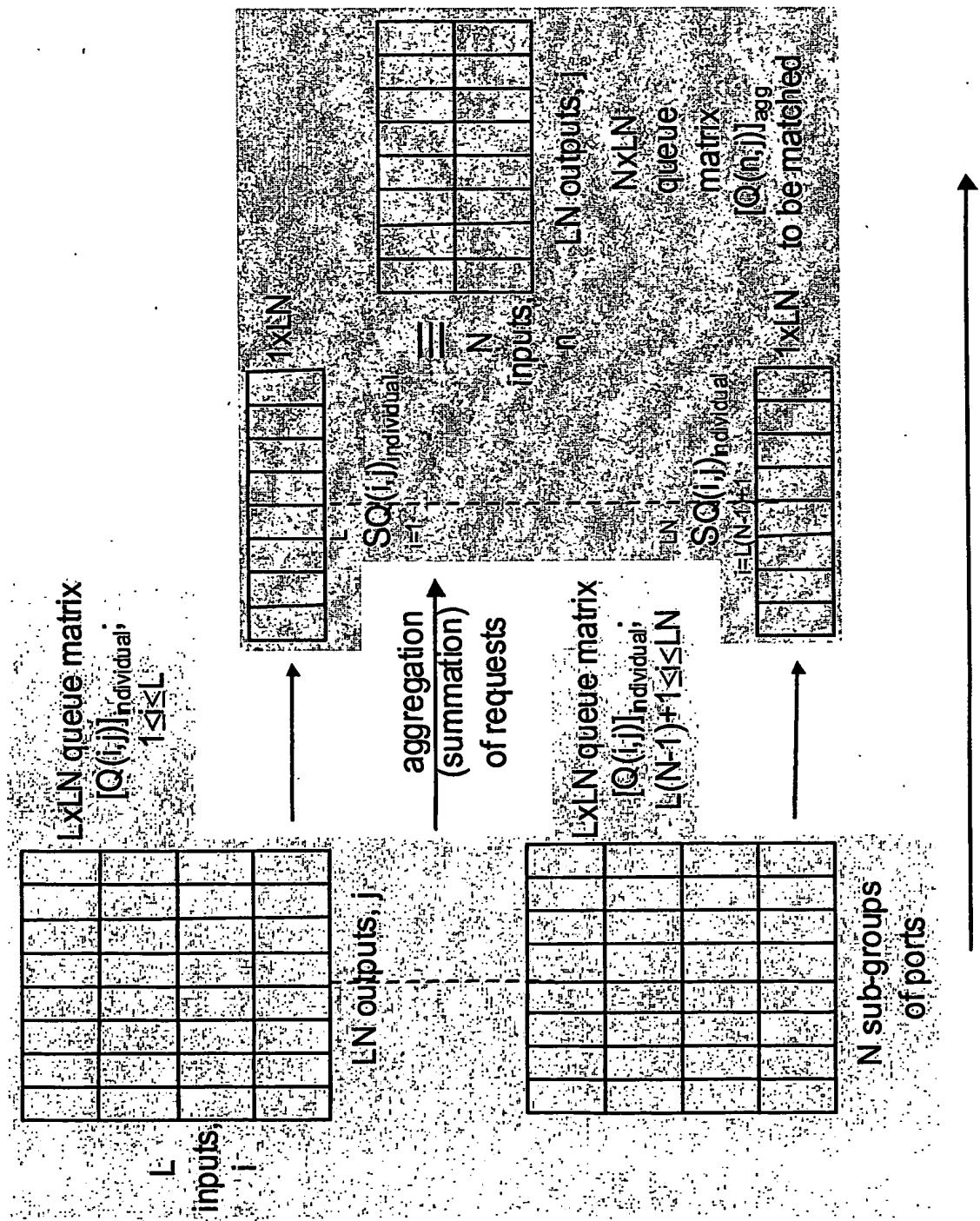


FIGURE 4

7/10



8/10

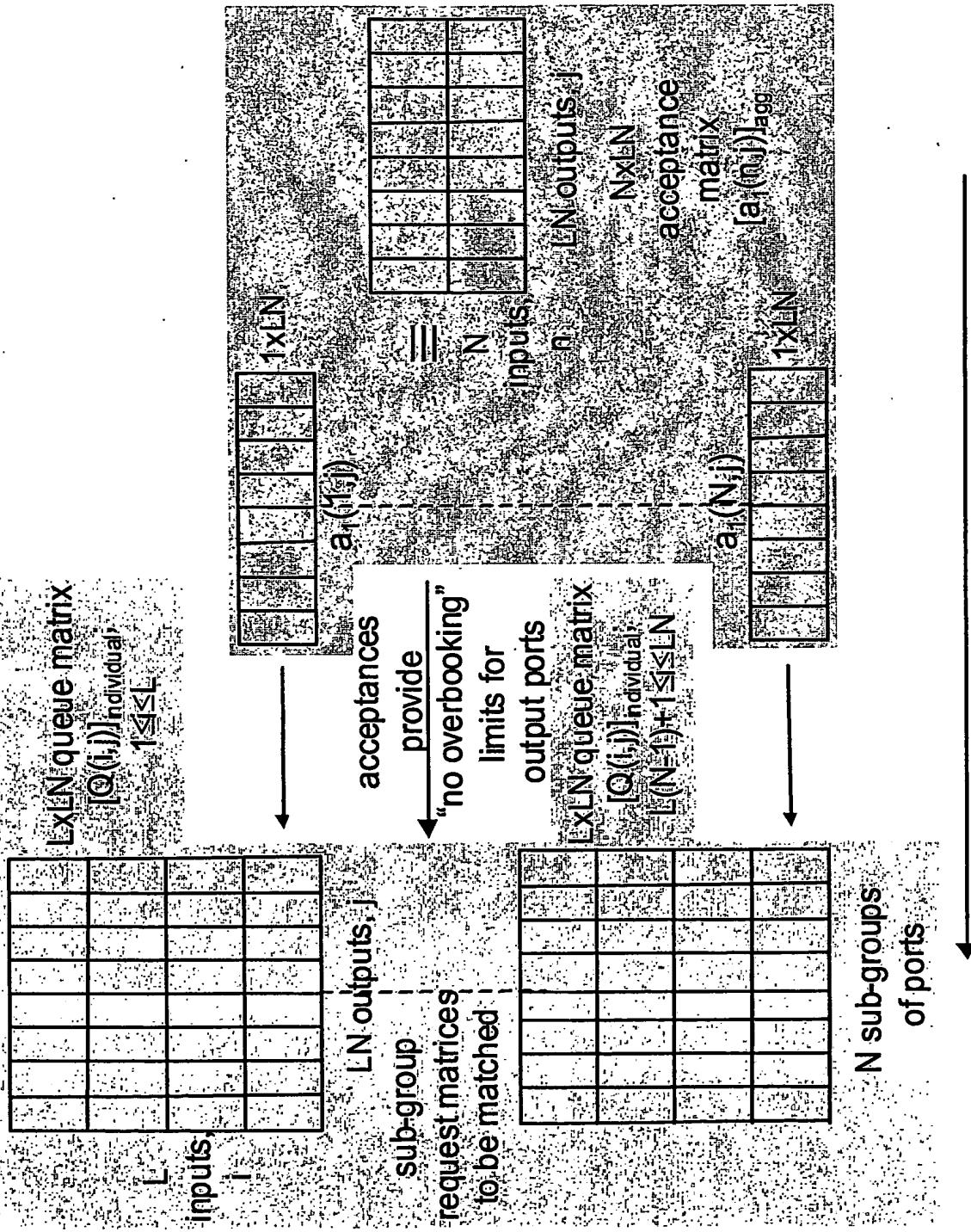


BEST AVAILABLE COPY

FIG 6

FIG 7

9/10



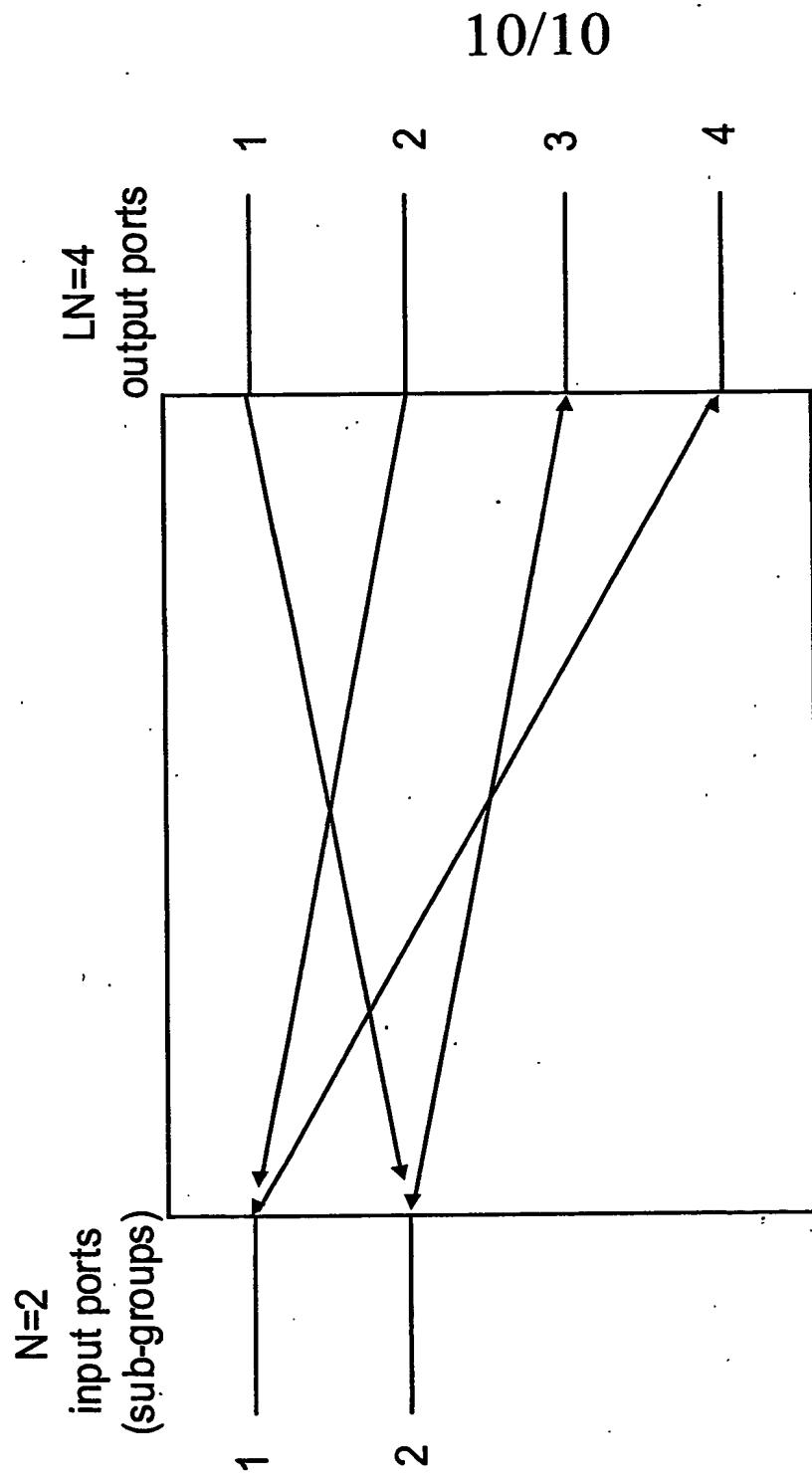


FIG 8